

Rational IBM Rational Developer for System z
バージョン 7.6.1

ホスト構成ガイド



Rational IBM Rational Developer for System z
バージョン 7.6.1

ホスト構成ガイド



— お願い —

本書をご使用になる前に、377 ページの『IBM Rational Developer for System z 資料に関する特記事項』に記載されている情報をお読みください。

本書は、IBM Rational Developer for System z バージョン 7.6.1 (プログラム番号 5724-T07)、および新しい版で明記されていない限り、これ以降のすべてのリリースおよびモディフィケーションに適用されます。

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原典： SC23-7658-04
Rational® IBM Rational Developer for System z
Version 7.6.1
Host Configuration Guide

発行： 日本アイ・ピー・エム株式会社

担当： トランスレーション・サービス・センター

第5版第1刷 2010.5

© Copyright IBM Corporation 2005, 2010.

目次

図	ix
---	----

表	xi
---	----

本書について	xiii
--------	------

本書の対象読者	xiii
変更の要約	xiii
文書内容の説明	xiv
計画	xiv
基本的なカスタマイズ	xiv
(オプション) 共通アクセス・リポジトリ・マネージャー (CARMA)	xv
(オプション) Application Deployment Manager	xv
(オプション) SCLM Developer Toolkit	xv
(オプション) その他のカスタマイズ・タスク	xvi
インストール検査	xvi
オペレーター・コマンド	xvi
構成問題のトラブルシューティング	xvi
セキュリティに関する考慮事項	xvii
Developer for System z について	xvii
WLM に関する考慮事項	xvii
チューニングに関する考慮事項	xvii
パフォーマンスに関する考慮事項	xvii
CICSTS に関する考慮事項	xvii
TSO 環境のカスタマイズ	xviii
複数のインスタンスの実行	xviii
マイグレーション・ガイド	xviii
SSL および X.509 認証のセットアップ	xviii
TCP/IP のセットアップ	xviii
INETD のセットアップ	xviii
APPC のセットアップ	xviii
必要条件	xviii

第 1 部 Developer for System z のカスタマイズ 1

第 1 章 計画 3

マイグレーションに関する考慮事項	3
計画に関する考慮事項	3
製品の概要	3
スキルの要件	4
時間の要件	4
インストール前の考慮事項	4
セットアップの選択	5
必要な製品	5
必要なリソース	6
構成前の考慮事項	9
ワークロード管理	9
リソース使用量とシステム限度	9
必要な製品の必須の構成	10

ユーザー ID に関する考慮事項	10
サーバーに関する考慮事項	11
構成方法	12
デプロイメント前の考慮事項	13
クライアント・チェックリスト	14

第 2 章 基本的なカスタマイズ 17

要件およびチェックリスト	17
カスタマイズのセットアップ	18
PARMLIB の変更	19
BPXPRMxx での z/OS UNIX 限度の設定	19
COMMNDxx への開始タスクの追加	20
LPALSTxx での LPA 定義	20
PROGxx での APF 許可	21
PROGxx での LINKLIST 定義	21
必要な LINKLIST 定義と LPA 定義	22
他の製品用の LINKLIST 定義	23
PROCLIB の変更	24
JES ジョブ・モニター	24
RSE デーモン	25
ロック・デーモン	26
PARM 変数の JCL での制限	26
ELAXF* リモート・ビルド・プロシージャ	27
セキュリティ定義	29
FEJJCNFG、JES ジョブ・モニター構成ファイル	30
rsed.envvars、RSE 構成ファイル	35
RSE サーバーに使用可能な PORTRANGE の定義	45
_RSE_JAVAOPTS での追加 Java 始動パラメータの定義	46
_RSE_CMDSERV_OPTS での追加 Java 始動パラメータの定義	51
ISPF.conf、ISPF の TSO/ISPF クライアント・ゲートウェイ構成ファイル	51
オプションのコンポーネント	53
インストール検査	53

第 3 章 (オプション) 共通アクセス・リポジトリ・マネージャー (CARMA) 55

要件およびチェックリスト	55
CARMA コンポーネント	56
CARMA VSAM のマイグレーションに関する注	57
CARMA への RSE インターフェース	58
バッチ実行依頼を使用した CARMA サーバーの始動	60
CRASRV.properties の調整	60
CRASUBMT の調整	61
(オプション) CRASTART を使用した代替の CARMA サーバーの始動	62
CRASRV.properties の調整	62
crastart.conf の調整	62
(オプション) TSO/ISPF クライアント・ゲートウェイを使用した代替の CARMA サーバーの始動	65

CRASRV.properties の調整	65
ISPF.conf の調整	65
(オプション) サンプルの Repository Access Manager	
(RAM) のアクティブ化	67
PDS RAM のアクティブ化	67
SCLM RAM のアクティブ化	68
スケルトン RAM のアクティブ化	68
(オプション) CA Endeavor SCM RAM のアクティブ	
化	68
要件およびチェックリスト	69
CA Endeavor SCM RAM の定義	69
バッチ実行依頼を使用した CA Endeavor SCM	
RAM の始動	70
CRASTART を使用した CA Endeavor SCM RAM	
の始動	72
(オプション) CRANDVRA のカスタマイズ	74
(オプション) CA Endeavor SCM RAM のカスタマ	
イズ	75
(オプション) 複数の RAM のサポート	75
例	76
(オプション) IRXJCL と CRAXJCL	77
CRAXJCL の作成	77

第 4 章 (オプション) Application

Deployment Manager 79

要件およびチェックリスト	79
CRD リポジトリ	80
CICS 管理ユーティリティ	81
RESTful と Web サービス	81
RESTful インターフェースを使用する CRD サーバ	
ー	82
CICS 主接続領域	82
CICS 非主接続領域	82
(オプション) CRD サーバー・トランザクション	
ID のカスタマイズ	83
Web サービス・インターフェースを使用する CRD	
サーバー	83
パイプライン・メッセージ・ハンドラー	84
CICS 主接続領域	85
CICS 非主接続領域	85
(オプション) マニフェスト・リポジトリ	86

第 5 章 (オプション) SCLM Developer

Toolkit 87

要件およびチェックリスト	87
前提条件	88
SCLMDT 用の ISPF.conf の更新	88
SCLMDT 用の rsed.envvars の更新	89
(オプション) ロング/ショート・ネーム変換	90
LSTRANS.FILE (ロング/ショート・ネーム変換	
VSAM) の作成	90
ロング/ショート・ネーム変換用の rsed.envvars の	
更新	92
(オプション) Ant のインストールおよびカスタマイ	
ズ	92
SCLMDT 用の SCLM の更新	93

WORKAREA からの古いファイルの除去	94
-----------------------	----

第 6 章 (オプション) その他のカスタマ

イズ・タスク 95

(オプション) DB2 ストアード・プロシージャ	95
ワークロード・マネージャー (WLM) の変更	95
PROCLIB の変更	96
DB2 の変更	97
(オプション) エンタープライズ・サービス・ツール	
(EST) サポート	97
(オプション) CICS 双方向言語サポート	98
(オプション) 診断用 IRZ エラー・メッセージ	99
(オプション) RSE SSL 暗号化	100
(オプション) RSE トレース	102
(オプション) ホスト・ベース・プロパティ・グル	
ープ	104
(オプション) ホスト・ベース・プロジェクト	105
(オプション) File Manager Integration	106
(オプション) 編集不可能文字	108
(オプション) REXEC (または SSH) の使用	109
z/OS UNIX サブプロジェクト用のリモート (ホ	
スト・ベース) アクション	109
代替 RSE 接続方式	110
REXEC (または SSH) のセットアップ	110
(オプション) TSO コマンド・サービス用の APPC	
トランザクション	111
準備	111
実装	113
APPC の使用に関する考慮事項	114
(オプション) WORKAREA クリーンアップ	115

第 7 章 インストール検査 117

開始タスクの検査	117
JMON、JES ジョブ・モニター	117
LOCKD、ロック・デーモン	117
RSED、RSE デーモン	117
サービスの検査	120
IVP の初期化	120
ポート可用性	121
TCP/IP のセットアップ	122
RSE デーモン接続	123
JES ジョブ・モニター接続	123
ロック・デーモン接続	124
ISPF の TSO/ISPF クライアント・ゲートウェイ	
接続	124
(オプション) APPC を使用した TSO コマン	
ド・サービス接続	126
(オプション) SCLMDT 接続	126
(オプション) REXEC 接続	128
(オプション) REXEC/SSH シェル・スクリプト	129

第 2 部 Developer for System z

情報 131

第 8 章 オペレーター・コマンド . . . 133

Start (S)	133
JES ジョブ・モニター	133
RSE デーモン	134
ロック・デーモン	134
Modify (F)	135
JES ジョブ・モニター	135
RSE デーモン	136
ロック・デーモン	139
Stop (P)	140
コンソール・メッセージ	140
JES ジョブ・モニター	140
RSE デーモン、RSE スレッド・プール・サーバ、およびロック・デーモン	140
構文図の読み方	143
記号	143
オペランド	143
構文例	144
非英数字およびブランク・スペース	144
複数のオペランドの選択	144
1 行より長い場合	144
構文フラグメント	144
第 9 章 構成問題のトラブルシューティング	145
FEKLOGS を使用したログとセットアップの分析	145
ログ・ファイル	146
JES ジョブ・モニター・ロギング	147
ロック・デーモン・ロギング	147
RSE デーモンおよびスレッド・プールのロギング	148
RSE ユーザー・ロギング	148
Fault Analyzer Integration ロギング	150
File Manager Integration ロギング	150
SCLM Developer Toolkit のロギング	150
CARMA ロギング	150
APPC トランザクション (TSO コマンド・サービス) ロギング	151
fekfivpi IVP テスト・ロギング	152
fekfivps IVP テスト・ロギング	152
ダンプ・ファイル	152
MVS ダンプ	152
Java ダンプ	153
z/OS UNIX ダンプ・ロケーション	154
トレース	154
JES ジョブ・モニターのトレース	154
RSE トレース	155
ロック・デーモンのトレース	156
CARMA トレース	156
エラー・フィードバック・トレース	156
z/OS UNIX 許可ビット	157
SETUID ファイル・システム属性	157
プログラム制御許可	158
APF 許可	159
スティッキー・ビット	160
予約済み TCP/IP ポート	161
アドレス・スペース・サイズ	162

始動 JCL の要件	162
SYS1.PARMLIB(BPXPRMxx) で設定される制限	162
セキュリティ・プロファイル内に保管される制限	163
システム出口によって強制される制限	163
64 ビット・アドレッシングでの制限	163
APPC トランザクションおよび TSO コマンド・サービス	164
各種情報	165
システム限度	165
必要条件に関する既知の問題	166
Host Connect Emulator	167

第 10 章 セキュリティーに関する考慮

事項 169

認証方式	170
ユーザー ID およびパスワード	170
ユーザー ID およびワнтаイム・パスワード	170
X.509 証明書	170
JES ジョブ・モニターの認証	171
接続セキュリティ	171
指定したポートのみに限定した外部通信	171
SSL を使用した通信暗号化	172
Port Of Entry 検査	172
TCP/IP ポート	173
外部通信	173
内部通信	174
CARMA と TCP/IP ポート	174
PassTicket の使用	175
監査ロギング	176
監査制御	176
監査データ	177
JES セキュリティー	177
ジョブに対するアクション - ターゲットの制限	177
ジョブに対するアクション - 実行の制限	179
スプール・ファイルへのアクセス	180
SSL 暗号化通信	181
X.509 証明書を使用したクライアント認証	182
認証局 (CA) の妥当性検査	183
(オプション) 証明書失効リスト (CRL) に対する照会	184
使用しているセキュリティ・ソフトウェアによる認証	184
RSE デーモンによる認証	185
Port Of Entry (POE) 検査	186
CICSTS セキュリティー	187
CRD リポジトリ	187
CICS トランザクション	187
SSL 暗号化通信	187
SCLM セキュリティー	187
Developer for System z 構成ファイル	188
JES ジョブ・モニター - FEJJCENFG	188
RSE - rsed.envvars	188
RSE - ssl.properties	189
セキュリティ定義	189
要件およびチェックリスト	190

セキュリティの設定およびクラスのアクティブ化	191
Developer for System z ユーザーの OMVS セグメントの定義	192
データ・セット・プロファイルの定義	192
Developer for System z 開始タスクの定義	196
JES コマンド・セキュリティの定義	197
セキュアな z/OS UNIX サーバーとしての RSE の定義	199
RSE の MVS プログラム制御ライブラリーの定義	199
RSE のアプリケーション保護の定義	200
RSE サーバーの PassTicket サポートの定義	200
RSE 用の z/OS UNIX プログラム制御ファイルの定義	201
セキュリティ設定の検査	202

第 11 章 Developer for System z について 205

コンポーネントの概要	205
Java アプリケーションとしての RSE	207
タスク所有者	208
接続のフロー	210
ロック・デーモン	212
ロックの解放	213
z/OS UNIX ディレクトリー構造	214
非システム管理者の更新特権	215

第 12 章 WLM に関する考慮事項 217

ワークロード分類	217
分類規則	218
目標の設定	219
目標の選択に関する考慮事項	220
STC	221
OMVS	222
JES	223
ASCH	224
CICS	225

第 13 章 チューニングに関する考慮事項 227

リソース使用量	227
概説	228
アドレス・スペースの数	229
プロセスの数	232
スレッドの数	235
ストレージの使用量	239
Java ヒープ・サイズの限度	239
アドレス・スペース・サイズの限度	239
サイズ見積もりのガイドライン	240
ストレージ使用量分析のサンプル	240
z/OS UNIX ファイル・システム・スペースの使用量	244
主要なリソース定義	247
/etc/rdz/rsed.envvars	247
SYS1.PARMLIB(BPXPRMxx)	248

さまざまなリソース定義	250
サーバー JCL での EXEC カード	250
FEK.#CUST.PARMLIB(FEJCNFG)	251
SYS1.PARMLIB(IEASYSxx)	251
SYS1.PARMLIB(IVTPRMxx)	251
SYS1.PARMLIB(ASCHPMxx)	252
モニター	252
RSE のモニター	252
z/OS UNIX のモニター	253
ネットワークのモニター	255
z/OS UNIX ファイル・システムのモニター	256
サンプル・セットアップ	256
スレッド・プールの数	256
最小限度の特定	257
限度の定義	257
リソース使用量のモニター	258

第 14 章 パフォーマンスに関する考慮事項 261

zFS ファイル・システムの使用	261
STEPLIB の使用の回避	261
システム・ライブラリーへのアクセスの改善	262
言語環境プログラム (LE) ランタイム・ライブラリー	262
アプリケーション開発	262
セキュリティ検査のパフォーマンスの向上	263
ワークロード管理	263
固定 Java ヒープ・サイズ	264
Java -Xquickstart オプション	264
JVM 間でのクラス共用	264
クラス共用の有効化	265
キャッシュ・サイズ制限	265
キャッシュ・セキュリティ	265
SYS1.PARMLIB(BPXPRMxx)	266
ディスク・スペース	266
キャッシュ管理ユーティリティー	266

第 15 章 CICS に関する考慮事項 269

RESTful と Web サービス	270
主接続領域と非主接続領域	270
CICS リソース・インストール・ロギング	271
Application Deployment Manager セキュリティー	271
CRD リポジトリ・セキュリティ	271
パイプライン・セキュリティ	271
トランザクション・セキュリティ	272
SSL 暗号化通信	273
リソース・セキュリティ	273
管理ユーティリティー	273
管理ユーティリティーのマイグレーションに関する注	277
管理ユーティリティーのメッセージ	278

第 16 章 TSO 環境のカスタマイズ 283

TSO コマンド・サービス	283
アクセス方式	283

TSO/ISPF クライアント・ゲートウェイ・アクセス 方式の使用	284
基本カスタマイズ - ISPF.conf	284
詳細設定 - 既存の ISPF プロファイルの使用	284
詳細設定 - 割り振り exec の使用	285
詳細設定 - 複数の割り振り exec の使用	285
詳細設定 - 複数の Developer for System z セッ トアップでの複数の ISPF.conf ファイル	286
APPC アクセス方式の使用	287
基本カスタマイズ - APPC トランザクション JCL	287
詳細設定 - 既存の ISPF プロファイルの使用	287
詳細設定 - 割り振り exec の使用	288
詳細設定 - 複数の Developer for System z セッ トアップでの複数の APPC トランザクション	288

第 17 章 複数のインスタンスの実行 291

シスプレックス全体での同一セットアップ	291
同一のソフトウェア・レベル、異なる構成ファイル	292
その他のすべての状態	293

第 18 章 マイグレーション・ガイド 297

マイグレーションに関する考慮事項	297
前に構成したファイルのバックアップ	297
I バージョン 7.6.1 のマイグレーションに関する注	299
バージョン 7.5 からバージョン 7.6 へのマイグレ ーション	301
IBM Rational Developer for System z, FMID HHOP760	301
構成可能なファイル	303
バージョン 7.1 からバージョン 7.5 へのマイグレ ーション	307
IBM Rational Developer for System z, FMID HHOP750	307
構成可能なファイル	309
バージョン 7.0 からバージョン 7.1 へのマイグレ ーション	311
IBM Rational Developer for System z, FMID HHOP710	311
IBM 共通アクセス・リポジトリ・マネージャ (CARMA)、FMID HCMA710	311
構成可能なファイル	312

付録 A. SSL および X.509 認証のセッ トアップ 315

秘密鍵と証明書を保管する場所の決定	316
RACF による鍵リングの作成	317
(オプション) 署名付き証明書の使用	318
既存の RSE セットアップのクローン作成	319
共存を可能にするための rsed.envvars の更新	319
SSL を有効にするための ssl.properties の更新	320
新しい RSE デーモンの作成による SSL のアクテ ィブ化	320
接続のテスト	321
(オプション) X.509 クライアント認証サポートの追 加	324

(オプション) gskkyman による鍵データベースの作 成	325
(オプション) keytool による鍵ストアの作成	327

付録 B. TCP/IP のセットアップ 329

ホスト名依存関係	329
リゾルバーについて	330
構成情報の検索順序について	330
z/OS UNIX 環境で使用される検索順序	331
基本リゾルバー構成ファイル	331
変換テーブル	332
ローカル・ホスト・テーブル	332
このセットアップ情報の Developer for System z へ の適用	333
ホスト・アドレスが正しく解決されない場合	336

付録 C. INETD のセットアップ 339

inetd.conf	339
ETC.SERVICES	341
z/OS UNIX 環境で使用される検索順序	341
ネイティブ MVS 環境で使用される検索順序	342
PROFILE.TCPIP ポート定義	342
/etc/inetd.pid	343
始動	343
/etc/rc	344
/etc/inittab	344
BPXBATCH	344
シェル・セッション	345
セキュリティ	345
Developer for System z の要件	347
INETD	347
REXEC (または SSH)	347

付録 D. APPC のセットアップ 349

VSAM	349
VTAM	350
SYS1.PARMLIB(APPCPMxx)	351
SYS1.PARMLIB(ASCHPMxx)	352
APPC の変更のアクティブ化	353
TSO コマンド・サービス・トランザクションの定 義	354
(オプション) 代替セットアップ・オプション	354
代替トランザクション名	354
複数の LU	354
LU セキュリティ	355

付録 E. 必要条件 357

z/OS ホストの前提条件	357
z/OS	357
SMP/E	359
SDK for z/OS Java 2 Technology Edition	359
z/OS ホストの相互前提条件	360
z/OS	360
COBOL コンパイラ	363
PL/I コンパイラ	363
Debug Tool for z/OS	363

CICS Transaction Server	364
IMS	365
DB2 for z/OS	365
Rational Team Concert for System z	366
File Manager	367
Fault Analyzer	367
REXX	367
Ported Tools	368
Ant	368
Endevor	368
参考文献.	369

参考資料	369
情報資料	371

用語集 373

IBM Rational Developer for System z 資料に関する特記事項 377

著作権使用許諾	378
商標	378

索引 381



1. JMON - JES ジョブ・モニター開始タスク	24	31. uchars.settings - 編集不可能文字構成ファイル	108
2. RSED - RSE デーモン開始タスク	25	32. APPC ISPF パネル用の REXX	112
3. LOCKD - ロック・デーモン開始タスク	26	33. START JMON オペレーター・コマンド	133
4. RSED - 代替 RSE デーモンの始動	27	34. START RSED オペレーター・コマンド	134
5. rsed.stdin.sh - 代替 RSE デーモンの始動	27	35. START LOCKD オペレーター・コマンド	134
6. FEJCNFG、JES ジョブ・モニター構成ファイル	31	36. MODIFY JMON オペレーター・コマンド	135
7. rsed.envvars - RSE 構成ファイル	37	37. MODIFY RSED オペレーター・コマンド	136
8. (続き)	38	38. MODIFY LOCKD オペレーター・コマンド	139
9. ISPF.conf - ISPF 構成ファイル	52	39. STOP オペレーター・コマンド	140
10. CRASRV.properties - CARMA 構成ファイル	58	40. TCP/IP ポート	173
11. CRASRV.properties - バッチ実行依頼を使用した CARMA の始動	60	41. コンポーネントの概要	205
12. CRASUBMT - バッチ実行依頼を使用した CARMA の始動	61	42. Java アプリケーションとしての RSE	207
13. CRASRV.properties - *CRASTART 代替 CARMA 始動	62	43. タスク所有者	209
14. crastart.conf - *CRASTART 代替 CARMA 始動	64	44. 接続のフロー	210
15. CRASRV.properties - *ISPF 代替 CARMA 始動	65	45. ロック・デーモンのフロー	212
16. ISPF.conf - *ISPF 代替 CARMA 始動	66	46. z/OS UNIX ディレクトリ構造	214
17. 図 x1. CRASRV.properties - バッチ実行依頼を使用した CA Endeavor SCM RAM の始動	70	47. WLM 分類	217
18. 図 x2. CRASUBMT - バッチ実行依頼を使用した CA Endeavor SCM RAM の始動	71	48. アドレス・スペースの最大数	231
19. 図 x3. CRASRV.properties - CRASTART を使用した CA Endeavor SCM RAM の始動	73	49. クライアントごとのアドレス・スペース数	232
20. crastart.conf - CRASTART を使用した CA Endeavor SCM RAM の始動	74	50. プロセスの最大数	234
21. SCLMDT 用の ISPF.conf の更新	89	51. クライアントごとのプロセス数	235
22. SCLMDT 用の rsed.envvars の更新	90	52. RSE スレッド・プール・スレッドの最大数	237
23. FLM02LST - ロング/ショート・ネーム変換セットアップ JCL	91	53. JES ジョブ・モニター・スレッドの最大数	237
24. ELAXMSAM - DB2 ストアード・プロシージャ・タスク	96	54. ログオン数 5 の場合のリソース使用量	241
25. ELAXMJCL - DB2 ストアード・プロシージャ・定義	97	55. ログオン数 5 の場合のリソース使用量 (続き)	242
26. ssl.properties - SSL 構成ファイル	101	56. PDS メンバー編集時のリソース使用量	243
27. rsecomm.properties - ロギング構成ファイル	103	57. z/OS UNIX ファイル・システム・スペースの使用量	245
28. propertiescfg.properties - ホスト・ベース・プロパティ・グループ構成ファイル	105	58. サンプル・セットアップのリソース使用量	259
29. projectcfg.properties - ホスト・ベース・プロジェクト構成ファイル	106	59. ADNJSAPU - CICSTS 管理ユーティリティ	275
30. FMIEXT.properties - File Manager 構成ファイル	107	60. FEKAPPCC - 2 番目の APPC トランザクション	289
		61. RSEDSSL - SSL 用の RSE デーモン・ユーザー・ジョブ	320
		62. 「ホスト証明書のインポート」ダイアログ	321
		63. 「設定」ダイアログ - 「SSL」	323
		64. INETD 始動 JCL	345
		65. APPC VSAM を作成するための JCL	350
		66. SYS1.SAMPLIB(ATBAPPL)	351
		67. SYS1.PARMLIB(APPCPMxx)	352
		68. SYS1.PARMLIB(ASCHPMxx)	353

表

1. 必要なリソース	6	26. セキュリティー・セットアップの変動要素	190
2. オプションのリソース	6	27. JES2 ジョブ・モニターのオペレーター・コマンド	198
3. 必須タスクに必要な管理者	7	28. JES3 ジョブ・モニターのオペレーター・コマンド	198
4. オプションのタスクに必要な管理者	8	29. WLM エントリー・ポイント・サブシステム	218
5. クライアント・チェックリスト - 必須部分	14	30. WLM 作業修飾子	219
6. クライアント・チェックリスト - オプション部分	14	31. WLM ワークロード	220
7. サンプル ELAXF* プロシージャ	27	32. WLM ワークロード - STC	221
8. ELAXF* 高位修飾子チェックリスト	29	33. WLM ワークロード - OMVS	222
9. LIMIT_COMMANDS コマンドの許可のマトリックス	33	34. WLM ワークロード - JES	224
10. crastart.conf の変数	64	35. WLM ワークロード - ASCH	224
11. デフォルトの CRD サーバー・トランザクション ID	83	36. WLM ワークロード - CICS	225
12. デフォルトの CRD サーバー・トランザクション ID	84	37. 共通のリソース使用量	228
13. SCLM 管理者チェックリスト	94	38. ユーザーごとに必要なリソース使用量	228
14. SSL 証明書の保管メカニズム	100	39. ユーザーごとのリソース使用量	229
15. 有効な鍵ストアのタイプ	102	40. アドレス・スペースの数	229
16. APPC トランザクション・チェックリスト	112	41. アドレス・スペースの限度	232
17. サービス用の IVP	120	42. プロセスの数	232
18. スレッド・プールのエラー状況	137	43. プロセスの限度	235
19. RSE コンソール・メッセージ	140	44. スレッドの数	236
20. JAVA_DUMP_TDUMP_PATTERN 変数	153	45. スレッドの限度	238
21. JES ジョブ・モニターのコンソール・コマンド	178	46. ログ出力ディレクティブ	246
22. LIMIT_COMMANDS コマンドの許可のマトリックス	178	47. バージョン 7.6 のカスタマイズ	304
23. 拡張 JESSPOOL プロファイル	179	48. バージョン 7.5 のカスタマイズ	309
24. LIMIT_VIEW のブラウザ権限のマトリックス	180	49. バージョン 7.1 のカスタマイズ	312
25. SSL 証明書の保管メカニズム	181	50. SSL 証明書の保管メカニズム	316
		51. リゾルバーで使用可能なローカル定義	335
		52. 参考資料	369
		53. 参照される Web サイト	371
		54. 情報資料	371

本書について

本書では、IBM Rational Developer for System z の機能の構成について説明しています。ここには、ご使用の z/OS® ホスト・システム上に IBM Rational Developer for System z バージョン 7.6.1 を構成する方法が記載されています。

これ以降、本書では以下の名前が使用されています。

- *IBM Rational Developer for System z* は *Developer for System z* と呼ばれます。
- 共通アクセス・リポジトリ・マネージャー は、*CARMA* と省略されます。
- *Software Configuration and Library Manager Developer Toolkit* は *SCLM Developer Toolkit* と呼ばれ、*SCLMDT* と省略されます。
- *z/OS UNIX*® システム・サービス は、*z/OS UNIX* と呼ばれます。
- 顧客情報管理システム (*CICS*) *Transaction Server* は、*CICSTS* と呼ばれ、*CICS*® と略されます。

IBM WebSphere Developer for System z、IBM WebSphere Developer for zSeries、および IBM® WebSphere Studio Enterprise Developer をはじめとする以前のリリースについては、当該リリースのホスト構成ガイドおよびプログラム・ディレクトリーに記載されている構成情報を使用してください。

本書の対象読者

本書は、IBM Rational Developer for System z バージョン 7.6.1、FMID HHOP760 を z/OS ホスト・システムにインストールおよび構成しようとするシステム・プログラマー向けに書かれています。

ここには、製品の完全セットアップを行うために必要となるさまざまなステップが、デフォルト以外のシナリオも含め、詳細にリストされています。本書を使用するには、z/OS UNIX システム・サービスおよび MVS ホスト・システムに精通している必要があります。

変更の要約

ここでは、「*Rational® Developer for System z*® バージョン 7.6.1 ホスト構成ガイド」(SC88-5663-02) (2010 年 5 月に更新) での変更点を要約します。

本文または図表に対して技術的な変更または追加が行われている場合には、その個所の左側に縦線を引いて示してあります。

本書には、「*Rational Developer for System z Version 7.6* ホスト構成ガイド」(SC88-5663-01) に記載されていた情報が含まれています。

新しい情報:

- 「*Rational Developer for System z v7.6 Host Configuration Release Notes*」に記載されている訂正と追加情報が盛り込まれています。

- バージョン 7.6.1 固有のマイグレーションに関する注。299 ページの『バージョン 7.6.1 のマイグレーションに関する注』を参照してください。
- 追加された文書概要。『文書内容の説明』を参照してください。
- 64 ビット Java™ のサポート。357 ページの『付録 E. 必要条件』を参照してください。
- ISPF パネルを通じた製品構成。9 ページの『構成前の考慮事項』を参照してください。
- rsed.envvars 内の新しいディレクティブ。35 ページの『rsed.envvars、RSE 構成ファイル』を参照してください。
- 新しい proclib メンバー。27 ページの『ELAXF* リモート・ビルド・プロシージャ』を参照してください。
- 新しい CARMA VSAM レイアウト。57 ページの『CARMA VSAM のマイグレーションに関する注』を参照してください。
- 複数の CARMA RAM のサポート。75 ページの『(オプション) 複数の RAM のサポート』を参照してください。
- Application Deployment Manager の新しいオプション。273 ページの『管理ユーティリティ』を参照してください。
- Application Deployment Manager の新しい VSAM レイアウト。277 ページの『管理ユーティリティのマイグレーションに関する注』を参照
- 新しいオペレーター・コマンド。133 ページの『第 8 章 オペレーター・コマンド』を参照してください。
- 新しいコンソール・メッセージ。140 ページの『コンソール・メッセージ』を参照
- ワークロード管理に関する情報。217 ページの『第 12 章 WLM に関する考慮事項』を参照してください。

文書内容の説明

ここでは、本書に記載されている情報を要約します。

計画

この章の情報を使用して、Developer for System z のインストールとデプロイメントを計画してください。

基本的なカスタマイズ

以下のカスタマイズ・ステップは、基本的な Developer for System z セットアップ用です。

- カスタマイズのセットアップ
- PARMLIB の変更
- PROCLIB の変更
- セキュリティー定義
- FEJCNFG、JES ジョブ・モニター構成ファイル
- rsed.envvars、RSE 構成ファイル

- ISPF.conf、ISPF の TSO/ISPF クライアント・ゲートウェイ構成ファイル

(オプション) 共通アクセス・リポジトリ・マネージャー (CARMA)

共通アクセス・リポジトリ・マネージャー (CARMA) は、Repository Access Manager (RAM) を作成する開発者向けの生産性援助機能です。RAM は、z/OS ベースの Software Configuration Manager (SCM) 用のアプリケーション・プログラミング・インターフェース (API) です。

次に、ユーザー作成アプリケーションは CARMA サーバーを始動でき、CARMA サーバーは RAM をロードし、SCM にアクセスする標準インターフェースを提供します。

IBM® Rational® Developer for System z Interface for CA Endevor® Software Configuration Manager は、Developer for System z クライアントが CA Endevor® SCM に直接アクセスできるようにします。

(オプション) Application Deployment Manager

Developer for System z は、さまざまなコンポーネントについて共通するデプロイメントの方法として、Application Deployment Manager の特定の機能を使用します。オプションのカスタマイズにより、より多くの Application Deployment Manager の機能が使用可能になり、以下のサービスを Developer for System z に追加できます。

- IBM CICS エクスプローラーは、CICS リソースを表示および管理するための Eclipse ベースのインフラストラクチャーを提供し、CICS ツール同士のさらに緊密な統合を可能にします。
- CICS リソース定義 (CRD) クライアントおよびサーバーは、以下の機能を提供します。
 - CICS リソース定義エディター
 - アプリケーション開発者が、CICS リソースを、制限付きで、制御されたセキュアな方法で定義できるようにします。
 - CICS 管理者がファイル定義内の物理データ・セット名属性を制御できるようにして、無許可または不正な VSAM データ・セットへの CICS 開発アクセスを防止します。
 - 各種の CICS 開発援助機能
 - 各種の CICS Web サービス開発援助機能

(オプション) SCLM Developer Toolkit

SCLM Developer Toolkit は、SCLM の機能を拡張するために必要なツールをクライアントに提供します。SCLM 自体はホスト・ベースのソース・コード・マネージャーであり、ISPF の一部として出荷されています。

SCLM Developer Toolkit は、Eclipse ベースのプラグインを備えており、SCLM へのインターフェースになります。また、レガシー・コード開発のすべての SCLM プロセスへのアクセスを提供するほか、メインフレーム上の SCLM と同期したワーク

ステーション上での完全な Java および J2EE 開発 (メインフレームからの J2EE コードのビルド、アセンブル、およびデプロイメントを含む) もサポートします。

(オプション) その他のカスタマイズ・タスク

このセクションは、さまざまなオプションのカスタマイズ・タスクを結合したものです。求めるサービスを構成するには、該当するセクションの説明に従ってください。

- DB2 ストアード・プロシージャ
- エンタープライズ・サービス・ツール (EST) サポート
- CICS 双方向言語サポート
- 診断用 IRZ エラー・メッセージ
- RSE SSL 暗号化
- RSE トレース
- ホスト・ベース・プロパティ・グループ
- ホスト・ベース・プロジェクト
- File Manager Integration
- 編集不可能文字
- REXEC (または SSH) の使用
- TSO コマンド・サービス用の APPC トランザクション
- WORKAREA クリーンアップ

インストール検査

製品のカスタマイズの完了後、この章で説明するインストール検査プログラム (IVP) を使用して、主要な製品コンポーネントのセットアップが正常であることを検査できます。

オペレーター・コマンド

この章では、Developer for System z で使用可能なオペレーター (またはコンソール) コマンドの概要を説明します。

構成問題のトラブルシューティング

この章は、Developer for System z の構成時に起きる可能性があるいくつかの一般的な問題について、ユーザーを支援するためのもので、以下のセクションで構成されています。

- ログとセットアップの分析に使用、FEKLOGS
- ログ・ファイル
- ダンプ・ファイル
- トレース
- z/OS UNIX 許可ビット
- 予約済み TCP/IP ポート
- アドレス・スペース・サイズ
- APPC トランザクションおよび TSO コマンド・サービス

- 各種情報

セキュリティに関する考慮事項

Developer for System z では、メインフレーム以外のワークステーション上にいるユーザーがメインフレームにアクセスできます。このため、接続要求の妥当性検査、ホストとワークステーション間のセキュアな通信の提供、およびアクティビティの許可と監査が、製品構成の重要な側面となります。

Developer for System z について

Developer for System z ホストは、クライアントがホスト・サービスとデータにアクセスできるようにするために相互に作用する、複数のコンポーネントで構成されています。これらのコンポーネントの設計を理解しておく、構成に関して適切な判断を行うことができます。

WLM に関する考慮事項

従来の z/OS アプリケーションとは異なり、Developer for System z は、ワークロード・マネージャー (WLM) で容易に識別できる一体構造のアプリケーションではありません。Developer for System z は、クライアントがホスト・サービスとデータにアクセスできるようにするために相互に作用する、複数のコンポーネントで構成されています。これらのサービスの一部は異なるアドレス・スペースでアクティブとなるため、WLM 分類も異なることになります。

チューニングに関する考慮事項

RSE (リモート・システム・エクスプローラー) は、Developer for System z の中核です。クライアントからの接続とワークロードを管理するために、RSE は、スレッド・プーリング・アドレス・スペースを制御するデーモン・アドレス・スペースから構成されています。デーモンは接続と管理のためのフォーカル・ポイントとして機能し、スレッド・プールはクライアント・ワークロードを処理します。

このため、RSE は Developer for System z セットアップをチューニングする場合の主要な対象となります。ただし、それぞれが 16 個以上のスレッドを使用する何百人ものユーザー、ある程度の大きさのストレージ、そして場合によっては 1 つ以上のアドレス・スペースを保守するには、Developer for System z と z/OS の両方を適切に構成する必要があります。

パフォーマンスに関する考慮事項

z/OS は高度にカスタマイズ可能なオペレーティング・システムであり、システムの場合によっては小さな) 変更で全体のパフォーマンスに大きな影響を与えることができます。この章では、Developer for System z のパフォーマンスを向上させるために行うことができるいくつかの変更を中心に説明します。

CICSTS に関する考慮事項

この章には、CICS Transaction Server 管理者に有益な情報が記載されています。

TSO 環境のカスタマイズ

この章は、Developer for System z で TSO 環境に DD ステートメントとデータ・セットを追加することにより、TSO ログオン・プロシーチャーを模倣するのに役立ちます。

複数のインスタンスの実行

同じシステム上で Developer for System z の複数のインスタンスをアクティブにしたい場合があります。例えば、アップグレードをテストするときなどです。しかし、TCP/IP ポートなど、一部のリソースは共用できないため、デフォルトが常に適用可能であるとは限りません。この章の情報を使用して Developer for System z の異なるインスタンスの共存を計画してください。その後、この構成ガイドを使用して、それらのインスタンスをカスタマイズできます。

マイグレーション・ガイド

ここでは、本製品の以前のリリースと比較したインストールおよび構成上の変更点に重点を置いて説明します。また、このリリースへのマイグレーションに関する一般的なガイドラインも示します。詳細については、本書内の関連するセクションを参照してください。

SSL および X.509 認証のセットアップ

この付録は、Secure Socket Layer (SSL) のセットアップ時、または既存のセットアップの検査時や変更時に起きる可能性があるいくつかの一般的な問題について、ユーザーを支援するためのものです。また、この付録には、X.509 証明書で自分自身を認証するユーザーをサポートするためのサンプルのセットアップも記載されています。

TCP/IP のセットアップ

この付録は、TCP/IP のセットアップ時、または既存のセットアップの検査時や変更時に起きる可能性があるいくつかの一般的な問題について、ユーザーを支援するためのものです。

INETD のセットアップ

この付録は、INETD のセットアップ時、または既存のセットアップの検査時や変更時に起きる可能性があるいくつかの一般的な問題について、ユーザーを支援するためのものです。INETD は、Developer for System z によって REXEC/SSH 機能のために使用されます。

APPC のセットアップ

この付録は、APPC (拡張プログラム間通信機能) のセットアップ時、または既存のセットアップの検査時や変更時に起きる可能性があるいくつかの一般的な問題について、ユーザーを支援するためのものです。

必要条件

この付録では、このバージョンの Developer for System z でのホストの前提条件および相互前提条件をリストします。

第 1 部 Developer for System z のカスタマイズ

第 1 章 計画

この章の情報を、357 ページの『付録 E. 必要条件』の情報とともに使用して、Developer for System z のインストールとデプロイメントを計画してください。

- 『マイグレーションに関する考慮事項』
- 『計画に関する考慮事項』
- 4 ページの『インストール前の考慮事項』
- 9 ページの『構成前の考慮事項』
- 13 ページの『デプロイメント前の考慮事項』
- 14 ページの『クライアント・チェックリスト』

マイグレーションに関する考慮事項

297 ページの『第 18 章 マイグレーション・ガイド』では、本製品の以前のリリースと比較したインストールおよび構成の変更点を説明します。この情報を使用して、Developer for System z の現行リリースへのマイグレーションを計画してください。

注:

1. 以前に IBM Rational Developer for System z、IBM WebSphere Developer for System z、IBM WebSphere Developer for zSeries、または IBM WebSphere Studio Enterprise Developer を使用していた場合は、IBM Rational Developer for System z バージョン 7.6.1 をインストールする「前」に、関連するカスタマイズ済みファイルを保存しておくことをお勧めします。カスタマイズする必要があるファイルの概要については、297 ページの『第 18 章 マイグレーション・ガイド』を参照してください。
2. Developer for System z の複数のインスタンスを実行する計画の場合は、291 ページの『第 17 章 複数のインスタンスの実行』を参照してください。

計画に関する考慮事項

製品の概要

Developer for System z は、ユーザーのパーソナル・コンピューターにインストールされたクライアント、および 1 台以上のホストにインストールされたサーバーで構成されます。本書では、z/OS システムであるホストを中心に説明します。ただし、AIX® や Linux® on System z などの他のオペレーティング・システムもサポートされています。

クライアントでは Eclipse ベースの開発環境が提供されます。この開発環境は、ホストとの統一されたグラフィカル・インターフェースを容易に実現し、特に、ホストからクライアントへの作業のオフロードを可能にしてホスト上のリソースを節約します。

ホスト部分は、永続的にアクティブな各種のタスクと、臨時に開始されるタスクから構成されます。これらのタスクによって、クライアントは z/OS ホストのさまざまなコンポーネント (MVS データ・セット、TSO コマンド、z/OS UNIX のファイルとコマンド、ジョブ実行依頼、ジョブ出力など) を処理することができます。

Developer for System z は、ホスト上のサブシステムやその他のアプリケーション・ソフトウェア (CICS、Debug Tool、Software Configuration Manager (SCM) など) と対話することもできます。そのためには、Developer for System z が対話を行うように構成されていることと、これらの (相互に必要な) 製品が使用可能であることが前提となります。

Developer for System z の設計に関する基礎知識については、205 ページの『第 11 章 Developer for System z について』を参照してください。

Developer for System z が提供する機能の詳細については、Developer for System z の Web サイト (<http://www-01.ibm.com/software/awdtools/rdz/>) を参照するか、IBM 担当員にお問い合わせください。

スキルの要件

Developer for System z ホストをインストールするには、SMP/E の最低限のスキルが必要です。

Developer for System z を構成するために必要なシステム・プログラミングの権限と専門知識は標準的なレベルを超えているため、他のユーザーからの支援が必要になる場合があります。7 ページの表 3、および 8 ページの表 4に、必須およびオプションのカスタマイズ・タスクに必要な管理者がリストされています。

時間の要件

Developer for System z ホスト・コンポーネントのインストールと構成にかかる時間は、以下のようなさまざまな要因に依存します。

- z/OS UNIX および TCP/IP の現行の構成
- 前提ソフトウェアと保守の可用性
- Developer for System z ユーザー用に OMVS セグメントが定義されているかどうか
- クライアントのインストールが完了しているユーザーの有無 (インストール済み環境をテストして発生のある可能性がある問題を報告するため)

経験則から判断すると、Developer for System z ホストをインストールして構成するプロセスには、完了までに 1 日から 4 日が必要です。これは、経験を積んだシステム・プログラマーが新規インストールを行う場合の時間要件です。問題が発生したり、必要とされるスキルが不足していたりする場合は、セットアップにさらに時間がかかります。

インストール前の考慮事項

製品の SMP/E インストールの詳細な手順については、「*IBM Rational Developer for System z Program Directory*」(GI88-4172) を参照してください。

注: Developer for System z がインストールされているファイル・システム (HFS または zFS) は、SETUID 許可ビットをオン (これはデフォルトです) にしてマウントする必要があります。NOSETUID パラメーターを指定してファイル・システムをマウントすると、ユーザーのセキュリティー環境が Developer for System z によって作成されず、クライアントの接続要求が失敗します。

Developer for System z の複数のインスタンスを実行する計画の場合は、291 ページの『第 17 章 複数のインスタンスの実行』を参照してください。

セットアップの選択

Developer for System z では、TSO コマンド・サービスへのアクセス方法を選択できます。ここで行った選択は、前提条件の必須の構成に影響します。以下のいずれかの方法を選択し、構成する必要があります。

- ISPF の TSO/ISPF クライアント・ゲートウェイ・サービス。これは最小の ISPF サービス・レベルを必要とします。これが、提供されたサンプルで使用されるデフォルトの方式です。
- APPC トランザクション (バージョン 7.1 より前のリリースと同様)

注: ISPF の TSO/ISPF クライアント・ゲートウェイは SCLM Developer Toolkit によっても使用され、オプションとして、共通アクセス・リポジトリ・マネージャー (CARMA) 用の代替の始動方式によっても使用されます。

必要な製品

357 ページの『付録 E. 必要条件』には、Developer for System z が機能する前にインストールされて操作可能になっていなければならない前提ソフトウェアのリストが示されています。また、Developer for System z の特定のフィーチャーをサポートするために相互に必要となるソフトウェアのリストもあります。該当するフィーチャーを設計どおりに機能させるには、これらの必要なソフトウェアをインストールし、実行時に操作可能になるようにしておく必要があります。

ご使用のバージョンの Developer for System z での、前提条件の製品と相互前提条件の製品の最新リストについては、Developer for System z オンライン・ライブラリー (<http://www-01.ibm.com/software/awdtools/rdz/library/>) の「*Rational Developer for System z Prerequisites*」(SC23-7659) を参照してください。(日本語版: 「Rational Developer for System z 前提条件」(SC88-4704) も出版されています)。ご使用のサイトのポリシーによっては、そのために少し時間を要する場合もあります。これらの必要な製品が使用可能になるように、事前に計画を立ててください。次に、基本セットアップの主要な必要条件を示します。

- z/OS 1.8 以上
- ISPF APAR OA29489 (TSO/ISPF クライアント・ゲートウェイ)
- Java 5.0 以上

注: 64 ビット版 Java を使用する場合は、Developer for System z APAR PM07305 用の PTF を適用する必要があります。この PTF は、Developer for System z 推奨サービスのページ (<http://www-01.ibm.com/support/docview.wss?rs=2294&context=SS2QJ2&uid=swg27006335>) で入手できます。

必要なリソース

Developer for System z を使用するには、表 1 にリストしたシステム・リソースの割り振りが必要です。オプションのサービスには、表 2 にリストしたリソースが必要です。ご使用のサイトのポリシーによっては、これらのリソースの取得にある程度の時間がかかる場合があるので、事前に計画を立ててリソースを使用できるようにしておいてください。

注: Developer for System z は、相互に、そしてクライアントと通信する複数のタスクから構成されています。これらのタスクでは、さまざまなタイマーを使用してパートナーとの通信が失われたことを検出します。このため、CPU の負荷が高いシステムや、Developer for System z に対するワークロード管理 (WLM) の設定が正しくないシステムでは、タイムアウトの問題が発生する可能性があります (タイムアウト・ウィンドウ内で CPU 時間が不足するため)。

表 1. 必要なリソース

リソース	デフォルト値	情報
APF 許可データ・セット	FEK.SFEKAUTH	21 ページの『PROGxx での APF 許可』
開始タスク	JMON、RSED、および LOCKD	11 ページの『サーバーに関する考慮事項』
ホスト限定使用のポート	6715	30 ページの『FEJCNFG、JES ジョブ・モニター構成ファイル』
ホスト限定使用のポート	4036	35 ページの『rsed.envvars、RSE 構成ファイル』
クライアント/ホスト通信用のポート	4035	24 ページの『PROCLIB の変更』
クライアント/ホスト通信用のポート範囲	使用可能な任意のポートを使用	45 ページの『RSE サーバーに使用可能な PORTRANGE の定義』
アプリケーション・セキュリティ定義	FEKAPPL に対する汎用アクセス READ	200 ページの『RSE のアプリケーション保護の定義』
PassTicket セキュリティ定義	デフォルトなし	200 ページの『RSE サーバーの PassTicket サポートの定義』

表 2. オプションのリソース

リソース	デフォルト値	情報
LINKLIST データ・セット	FEK.SFEKAUTH および FEK.SFEKLOAD	87 ページの『第 5 章 (オプション) SCLM Developer Toolkit』
LPA データ・セット	FEK.SFEKLPA	55 ページの『第 3 章 (オプション) 共通アクセス・リポジトリ・マネージャー (CARMA)』

表 2. オプションのリソース (続き)

リソース	デフォルト値	情報
ホスト限定使用のポート範囲	5227 から 5326 (100 ポート)	55 ページの『第 3 章 (オプション) 共通アクセス・リポジトリ・マネージャー (CARMA)』
ホスト限定使用のポート	使用可能な任意のポートを使用	111 ページの『(オプション) TSO コマンド・サービス用の APPC トランザクション』
クライアント/ホスト通信用のポート	デフォルトなし	79 ページの『第 4 章 (オプション) Application Deployment Manager』
CICS CSD アップデート	複数値	79 ページの『第 4 章 (オプション) Application Deployment Manager』
CICS JCL アップデート	FEK.SFEKLOAD	<ul style="list-style-type: none"> 79 ページの『第 4 章 (オプション) Application Deployment Manager』 98 ページの『(オプション) CICS 双方向言語サポート』

Developer for System z を構成するために必要なシステム・プログラミングの権限と専門知識は標準的なレベルを超えているため、他のユーザーからの最低限の支援が必要になる場合があります。表 3、および 8 ページの表 4に、必須およびオプションのカスタマイズ・タスクに必要な管理者がリストされています。

表 3. 必須タスクに必要な管理者

管理者	タスク	情報
システム	すべてのカスタマイズ・タスクには、一般的なシステム・プログラマーのアクションが必要である	適用外

表 3. 必須タスクに必要な管理者 (続き)

管理者	タスク	情報
セキュリティ	<ul style="list-style-type: none"> • Developer for System z ユーザーの OMVS セグメントを定義する • データ・セット・プロファイルを定義する • 開始タスクを定義する • オペレーター・コマンド・セキュリティを定義する • z/OS UNIX サーバー・プロファイルを定義する • アプリケーション・セキュリティを定義する • PassTicket サポートを定義する • プログラム制御データ・セットを定義する • プログラム制御 z/OS UNIX ファイルを定義する 	169 ページの『第 10 章 セキュリティに関する考慮事項』
TCP/IP	新しい TCP/IP ポートを定義する	173 ページの『TCP/IP ポート』
WLM	開始タスクの目標をサーバーとその子プロセスに割り当てる	217 ページの『第 12 章 WLM に関する考慮事項』

表 4. オプションのタスクに必要な管理者

管理者	タスク	情報
システム	すべてのカスタマイズ・タスクには、一般的なシステム・プログラマーのアクションが必要である	適用外
セキュリティ	<ul style="list-style-type: none"> • データ・セット・プロファイルを定義する • プログラム制御データ・セットを定義する • xxx* ジョブを実行依頼するための許可を定義する • CICS トランザクション・セキュリティを定義する • SSL の証明書を追加する • X.509 クライアント証明書サポートを定義する 	<ul style="list-style-type: none"> • 169 ページの『第 10 章 セキュリティに関する考慮事項』 • 269 ページの『第 15 章 CICSTS に関する考慮事項』 • 315 ページの『付録 A. SSL および X.509 認証のセットアップ』
TCP/IP	新しい TCP/IP ポートを定義する	173 ページの『TCP/IP ポート』

表 4. オプションのタスクに必要な管理者 (続き)

管理者	タスク	情報
SCLM	<ul style="list-style-type: none"> • JAVA/J2EE サポート用の SCLM 言語変換プログラムを定義する • JAVA/J2EE サポート用の SCLM タイプを定義する 	87 ページの『第 5 章 (オプション) SCLM Developer Toolkit』
CICS TS	<ul style="list-style-type: none"> • CICS 領域 JCL を更新する • CICS 領域 CSD を更新する • CICS グループを定義する • CICS トランザクション名を定義する • CICS に対してプログラムを定義する 	<ul style="list-style-type: none"> • 79 ページの『第 4 章 (オプション) Application Deployment Manager』 • 98 ページの『(オプション) CICS 双方向言語サポート』
DB2®	DB2 ストアード・プロシージャを定義する	95 ページの『(オプション) DB2 ストアード・プロシージャ』
WLM	<ul style="list-style-type: none"> • DB2 ストアード・プロシージャに最終目標を割り当てる • APPC トランザクションに TSO のような最終目標を割り当てる 	<ul style="list-style-type: none"> • 95 ページの『(オプション) DB2 ストアード・プロシージャ』 • 217 ページの『第 12 章 WLM に関する考慮事項』
APPC	APPC トランザクションを定義する	111 ページの『(オプション) TSO コマンド・サービス用の APPC トランザクション』

構成前の考慮事項

ワークロード管理

従来の z/OS アプリケーションとは異なり、Developer for System z は、ワークロード・マネージャー (WLM) で容易に識別できる一体構造のアプリケーションではありません。Developer for System z は、クライアントがホスト・サービスとデータにアクセスできるようにするために相互に作用する、複数のコンポーネントで構成されています。217 ページの『第 12 章 WLM に関する考慮事項』を参照し、それに従って WLM 構成を計画してください。

リソース使用量とシステム限度

使用中の Developer for System z では、アドレス・スペースや z/OS UNIX のプロセスとスレッドなど、不定数のシステム・リソースが使用されます。これらのリソースの可用性は、さまざまなシステム定義によって制限されます。227 ページの

『第 13 章 チューニングに関する考慮事項』を参照して主要なリソースの使用量を見積もり、それに合わせてシステム構成を計画してください。

必要な製品の必須の構成

MVS システム・プログラマー、セキュリティー管理者、および TCP/IP 管理者に、必要な製品とソフトウェアがインストールされ、テストされ、機能しているかどうかを確認してください。必要であるのに見過ごしてしまいがちなカスタマイズ・タスクのいくつかを、以下に示します。

- すべての Developer for System z ユーザーが、Java ディレクトリーに対する READ アクセス権と EXECUTE アクセス権を持っている必要があります。
- すべての Developer for System z ユーザーが、/tmp/ ディレクトリーに対する READ、WRITE、および EXECUTE のアクセス権を持っている必要があります。
- z/OS UNIX サブプロジェクトのリモート (ホスト・ベースの) アクションを実行するには、ホスト上で z/OS UNIX バージョンの REXEC または SSH がアクティブであることが必要です。

ユーザー ID に関する考慮事項

Developer for System z ユーザーのユーザー ID には、以下の属性が (最低でも) 必要です。

- TSO アクセス (通常の領域サイズ)

注: インストール検査プログラム (IVP) を実行するユーザー ID には、大きな領域サイズが必要です。これは、大量のメモリーを必要とする機能 (Java など) が実行されるからです。領域サイズは、131072 キロバイト (128 メガバイト) 以上に設定してください。

- ユーザー ID とそのデフォルト・グループの両方に対して、セキュリティー・システム (RACF® など) に定義された OMVS セグメント
 - HOME フィールドは、ユーザー (READ、WRITE、および EXECUTE アクセス権を持つ) に割り振られたホーム・ディレクトリーを参照する必要があります。
 - OMVS セグメント内の PROGRAM フィールドは、/bin/sh とするか、または、それ以外の有効な z/OS UNIX シェル (/bin/tcsh など) にしてください。
 - ASSIZEMAX フィールドは設定せず、システム・デフォルトが使用されるようにしてください。
 - ユーザー ID は UID 0 を必要としません。

例 (コマンド **LISTUSER userid NORACF OMVS**):

USER=userid

```
OMVS INFORMATION
-----
UID= 0000003200
HOME= /u/userid
PROGRAM= /bin/sh
CPUTIMEMAX= NONE
ASSIZEMAX= NONE
```

```
FILEPROCMAX= NONE
PROCUSERMAX= NONE
THREADSMAX= NONE
MMAPAREAMAX= NONE
```

- ユーザー ID のデフォルト・グループには GID が必要です。

例 (コマンド **LISTGRP group NORACE OMVS**):

```
GROUP group
```

```
OMVS INFORMATION
-----
GID= 0000003243
```

- Developer for System z インストールおよび構成ディレクトリーとファイル (デフォルトでは /usr/lpp/rdz/*、/etc/rdz/*、および /var/rdz/*) への READ および EXECUTE アクセス権
- Developer for System z WORKAREA ディレクトリー (デフォルトでは /var/rdz/WORKAREA) への READ、WRITE、および EXECUTE アクセス権
- Developer for System z インストール・データ・セット (デフォルトでは FEK.SFEK*) への READ アクセス権

サーバーに関する考慮事項

Developer for System z は 3 つの永続的にアクティブなサーバーから構成され、これらのサーバーは、開始タスクまたはユーザー・ジョブとすることができます。これらのサーバーは、要求されたサービスをそれら自体が提供するか、他のサーバー (z/OS UNIX スレッドまたはユーザー・ジョブなど) を始動してサービスを提供します。

- JES ジョブ・モニター (JMON) は、JES に関連したすべてのサービスを提供します。
- ロック・デーモン (LOCKD) は、データ・セット・ロックのトラッキング・サービスを提供します。
- リモート・システム・エクスプローラー (RSE) は、クライアントをホストに接続したり、特定のサービス用に他のサーバーを始動するなどの、コア・サービスを提供します。RSE は、次の 2 つの論理エンティティーから構成されます。
 - RSE デーモン (RSED)。これは接続セットアップを管理し、単一サーバー・モードでの実行を担当します。
 - RSE サーバー。これは個々のクライアント要求を処理します。

JES ジョブ・モニター (JMON) は、JES に関連したすべてのサービスを提供します。

- JES ジョブ・モニターが使用するセキュリティ・メカニズムは、JES ジョブ・モニターが存在するデータ・セットがセキュアであることに依存しています。つまり、信頼されたシステム管理者のみがライブラリーと構成ファイルを更新できる状態でなければなりません。

リモート・システム・エクスプローラー (RSE) は、クライアントをホストに接続するなどのコア・サービスを提供する Developer for System z コンポーネントです。

- バージョン 7.5 以降、RSE デーモンは INETD 管理対象プロセスではなく、開始タスクになりました。

- バージョン 7.5 以降、RSE サーバーは単一サーバー・モデルを使用します。これに対し、以前のバージョンでは、それぞれのクライアント/ホスト接続ごとに専用の RSE サーバーがありました。
- さまざまなレベルの通信セキュリティが、RSE によってサポートされています。
 - 外部 (クライアント/ホスト) 通信を、指定したポートだけに制限できます。このフィーチャーは、デフォルトでは使用不可に設定されます。
 - 外部 (クライアント/ホスト) 通信を SSL で暗号化できます。このフィーチャーは、デフォルトでは使用不可に設定されます。
 - 信頼できる TCP/IP アドレスのみにアクセスを許可できるようにするために、Port Of Entry (POE) 検査を使用できます。このフィーチャーは、デフォルトでは使用不可に設定されます。
- RSE は、以下のような複数のクライアント認証方式もサポートしています。
 - ユーザー ID およびパスワード
 - ユーザー ID およびワンタイム・パスワード
 - X.509 証明書
- RSE が使用するセキュリティ・メカニズムは、RSE サーバーが存在するファイル・システムがセキュアであることに依存しています。つまり、信頼されたシステム管理者のみがライブラリーと構成ファイルを更新できる状態でなければなりません。

173 ページの『TCP/IP ポート』で説明されているように、特定のホスト・サービスと、したがって、それらのポートは、クライアントが接続するために使用可能でなければならず、また、ホストを保護しているファイアウォールに対して定義されていなければなりません。これ以外に Developer for System z が使用するポートは、すべて、ホスト専用トラフィックを持ちます。以下に、基本的な Developer for System z セットアップに必要なポートを示します。

- クライアント/ホスト通信セットアップ用 (TCP プロトコルを使用) の RSE デーモン、デフォルト・ポート 4035。
- クライアント/ホスト通信用 (TCP プロトコルを使用) の RSE サーバー。デフォルトでは、使用可能な任意のポートを使用できますが、これは指定された範囲に制限できます。

注: 以前のクライアント (バージョン 7.0 以前) は、JES ジョブ・モニター (TCP プロトコルを使用) と直接通信します。デフォルト・ポート 6715。

構成方法

バージョン 7.6.1 より、Developer for System z では、ISPF パネル・アプリケーションを使用する代替の方法で、製品のホスト側を構成できるようになりました。これにより、ユーザーは以下の方法から選択することができます。

- ISPF パネル・アプリケーションを使用する方法。このアプリケーションでは、必要なカスタマイズ・ステップおよび選ばれたオプションのカスタマイズ・ステップを順に実行していきます。詳細については、Developer for System z インターネット・ライブラリー (<http://www-306.ibm.com/software/awdtools/rdz/library/>) で提供されているホワイト・ペーパー「*Host Configuration Utility*」を参照してください。

- 「ホスト構成クイック・スタート・ガイド」を使用する方法。このガイドでは、必要なカスタマイズ・ステップが順に説明されています。このガイドの適用範囲は、基本セットアップに限定されています。
- 「ホスト構成ガイド」を使用する方法。このガイドでは、必要なカスタマイズ・ステップとすべてのオプションのカスタマイズ・ステップが順に説明されています。このガイドには、構成可能なすべてのオプションが、デフォルト以外のシナリオも含めて掲載されています。

デプロイメント前の考慮事項

Developer for System z では、インストールのクローンを異なるシステムに作成でき、各システム上で SMP/E のインストールを行わなくても済みます。

以下のデータ・セット、ディレクトリー、およびファイルは、他のシステムへのデプロイメントに必須です。ファイルを別のロケーションへコピーしてある場合は、下記のリスト内の相当するファイルをそのファイルに置き換える必要があります。

注: 以下のリストは、前提条件および相互前提条件のソフトウェアのデプロイメントの必要性を考慮したものではありません。

- FEK.SFEKAUTH(*)
- FEK.SFEKLOAD(*)
- FEK.SFEKPROC(*)
- FEK.#CUST.PARMLIB(*)
- FEK.#CUST.PROCLIB(*)
- /usr/lpp/rdz/*
- /etc/rdz/*
- /var/rdz/* (ディレクトリー構造のみ)
- オプションのパーツ:
 - FEK.SFEKLPA(*)
 - FEK.#CUST.CNTL(*)
 - FEK.#CUST.JCL でのカスタマイズ・ジョブで生成された定義、データ・セット、ファイル、およびディレクトリー

注:

1. FEK および /usr/lpp/rdz は、製品のインストール時に使用された高位修飾子およびパスです。FEK.#CUST、/etc/rdz、および /var/rdz は、製品のカスタマイズ時に使用されたデフォルトのロケーションです (詳細については、18 ページの『カスタマイズのセットアップ』を参照してください)。
2. 製品の z/OS UNIX の部分をデプロイしやすくするために、Developer for System z を専用ファイル・システム (HFS または zFS) にインストールしてください。
3. 専用ファイル・システムを使用できない場合は、z/OS UNIX の tar コマンドなどのアーカイブ・ツールを使用して z/OS UNIX ディレクトリーをシステム間で転送してください。これにより、Developer for System z のファイルとディレクトリーの属性 (プログラム制御など) が保存されます。

Developer for System z インストール・ディレクトリーをアーカイブおよび復元するための以下のサンプル・コマンドについて詳しくは、「*UNIX System Services コマンド解説書*」(SA88-8641) を参照してください。

- アーカイブ: `cd /SYS1/usr/lpp/rdz; tar -cSf /u/userid/rdz.tar`
- 復元: `cd /SYS2/usr/lpp/rdz; tar -xSf /u/userid/rdz.tar`

クライアント・チェックリスト

Developer for System z クライアントのユーザーは、クライアントを正しく機能させるために、特定のホスト・カスタマイズの結果 (TCP/IP ポート番号など) を把握しておく必要があります。これらのチェックリストを使用して、必要な情報を収集してください。

表 5 のチェックリストは、必須のカスタマイズ・ステップに必要な結果を示しています。表 6 は、オプションのカスタマイズ・ステップに必要な結果を示しています。

表 5. クライアント・チェックリスト - 必須部分

カスタマイズ	値
JES ジョブ・モニター・サーバーのポート番号 (デフォルトは 6715) 30 ページの『FEJJCNFG、JES ジョブ・モニター構成ファイル』の SERV_PORT を参照してください。	
RSE デーモンの TCP/IP ポート番号 (デフォルトは 4035) 25 ページの『RSE デーモン』を参照してください。	

表 6. クライアント・チェックリスト - オプション部分

カスタマイズ	値
ELAXF* プロシージャーのロケーション (システム・プロシージャー・ライブラリーに入っていない場合) 27 ページの『ELAXF* リモート・ビルド・プロシージャー』で JCLLIB に関する注を参照してください。	
ELAXF* プロシージャーのプロシージャー名またはステップ名 (変更された場合) 27 ページの『ELAXF* リモート・ビルド・プロシージャー』で、それらの変更に関する注を参照してください。	
DB2 ストアード・プロシージャー名 (デフォルトは ELAXMSAM) 291 ページの『第 17 章 複数のインスタンスの実行』で、DB2 ストアード・プロシージャーに関する情報を参照してください。	
DB2 ストアード・プロシージャーのロケーション (システム・プロシージャー・ライブラリーに入っていない場合) 95 ページの『(オプション) DB2 ストアード・プロシージャー』を参照してください。	

表 6. クライアント・チェックリスト - オプション部分 (続き)

カスタマイズ	値
<p>(相互前提条件) Host Connect Emulator の TN3270 ポート番号 (デフォルトは 23)</p> <p>169 ページの『第 10 章 セキュリティーに関する考慮事項』を参照してください。</p>	
<p>(相互前提条件) REXEC または SSH ポート番号 (デフォルトはそれぞれ 512 と 22)</p> <p>109 ページの『(オプション) REXEC (または SSH) の使用』を参照してください。</p>	
<p>REXEC/SSH 接続方式を使用した場合の server.zseries ファイルのロケーション (デフォルトは /etc/rdz)。</p> <p>109 ページの『(オプション) REXEC (または SSH) の使用』を参照してください。</p>	
<p>CARMA SCLM RAM データ・セット割り振りの CRA#ASLM JCL のロケーション (デフォルトは FEK.#CUST.JCL)。</p> <p>68 ページの『SCLM RAM のアクティブ化』で CRA#ASLM に関する注を参照してください。</p>	

第 2 章 基本的なカスタマイズ

以下のカスタマイズ・ステップは、基本的な Developer for System z セットアップ用です。オプション・コンポーネントのカスタマイズ要件については、それらのコンポーネントに関する章を参照してください。

要件およびチェックリスト

このカスタマイズ・タスクを完了するには、セキュリティー管理者および TCP/IP 管理者の支援が必要になります。このタスクには、以下のリソースと特殊なカスタマイズ・タスクが必要です。

- APF 許可データ・セット
- 各種の PARMLIB 更新
- 各種のセキュリティー・ソフトウェア更新
- 内部およびクライアント/ホスト通信用の各種 TCP/IP ポート

インストールを検証し、ご使用のサイトで Developer for System z の使用を開始するためには、以下のタスクを実行する必要があります。特に断りがない限り、すべてのタスクは必須です。

1. カスタマイズ可能なサンプルのコピーを作成し、Developer for System z 用の作業環境を作成します。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。
2. z/OS UNIX システム限度を更新し、開始タスクを開始します。さらに、APF 許可データ・セットと LINKLIST データ・セットを定義し、オプションで LPA データ・セットを定義します。詳細については、19 ページの『PARMLIB の変更』を参照してください。
3. 開始タスク・プロシーチャーを作成し、プロシーチャーをコンパイル/リンクします。詳細については、24 ページの『PROCLIB の変更』を参照してください。
4. セキュリティー定義を更新します。詳細については、29 ページの『セキュリティー定義』を参照してください。また、PassTicket を使用してスレッドのセキュリティーが確立される仕組みを認識し、理解しておく必要があります。詳細については、175 ページの『PassTicket の使用』を参照してください。
5. Developer for System z 構成ファイルをカスタマイズします。詳細については、以下を参照してください。
 - 30 ページの『FEJCNFG、JES ジョブ・モニター構成ファイル』
 - 35 ページの『rsed.envvars、RSE 構成ファイル』
 - 51 ページの『ISPF.conf、ISPF の TSO/ISPF クライアント・ゲートウェイ構成ファイル』

カスタマイズのセットアップ

Developer for System z には、いくつかのサンプル構成ファイルとサンプル JCL が添付されています。カスタマイズした内容を保守の適用時に上書きしてしまわないように、これらのメンバーと z/OS UNIX ファイルを別のロケーションへすべてコピーし、そのコピーをカスタマイズする必要があります。

Developer for System z の一部の機能は、z/OS UNIX 内に特定のディレクトリーが存在することを必要とします。それらのディレクトリーを製品のカスタマイズ時に作成する必要があります。インストールの労力を軽減するために、コピーと必要なディレクトリーを作成するサンプル・ジョブ、FEKSETUP が提供されています。

データ・セット FEK.SFEKSAMP 内のサンプル・メンバー FEKSETUP をカスタマイズおよび実行依頼して、構成ファイルおよび構成 JCL のカスタマイズ可能コピーを作成し、必要な z/OS UNIX ディレクトリーを作成します。必要なカスタマイズ・ステップは、このメンバー内に記述されています。

このジョブは、以下のタスクを実行します。

- FEK.#CUST.PARMLIB を作成し、これにサンプル構成ファイルを取り込みます。
- FEK.#CUST.PROCLIB を作成し、これにサンプル SYS1.PROCLIB メンバーを取り込みます。
- FEK.#CUST.JCL を作成し、これにサンプル構成 JCL を取り込みます。
- FEK.#CUST.CNTL を作成し、これにサンプルのサーバー始動スクリプトを取り込みます。
- FEK.#CUST.ASM を作成し、これにサンプル・アセンブラー・ソース・コードを取り込みます。
- FEK.#CUST.COBOL を作成し、これにサンプル COBOL ソース・コードを取り込みます。
- /etc/rdz/* を作成し、これにサンプルの構成ファイルを取り込みます。
- /var/rdz/* を各種の Developer for System z 機能の作業ディレクトリーとして作成します。

注:

1. この資料の構成ステップでは、特に断りがない限り、FEKSETUP ジョブによって作成されたメンバー/ファイル・ロケーションを使用します。更新してはならないオリジナルのサンプルは、FEK.SFEKSAMP および /usr/lpp/rdz/samples/ に入っています。
2. すべての Developer for System z z/OS UNIX ファイルを同じファイル・システム (HFS または zFS) に保持したいが、構成ファイルは /etc/rdz に置いておきたいという場合は、シンボリック・リンクを使用して、この問題を解決できます。以下のサンプル z/OS UNIX コマンドは、既存のファイル・システム (/usr/lpp/rdz/cust) 内に新規ディレクトリーを作成し、それへのシンボリック・リンク (/etc/rdz) を定義します。

```
mkdir /usr/lpp/rdz/cust
ln -s /usr/lpp/rdz/cust /etc/rdz
```

PARMLIB の変更

以下にリストした PARMLIB 定義の詳細については、「MVS 初期設定およびチューニング解説書」(SA88-8564)を参照してください。サンプルのコンソール・コマンドの詳細については、「MVS システム・コマンド」(SA88-8593)を参照してください。

BPXPRMxx での z/OS UNIX 限度の設定

リモート・システム・エクスプローラー (RSE) は、クライアントをホストに接続するなどのコア・サービスを提供する、z/OS UNIX ベースのプロセスです。したがって、同時にアクティブになる Developer for System z ユーザーの数およびそれらのユーザーの平均的なワークロードに基づいて、BPXPRMxx で z/OS UNIX システム限度に正しい値を設定することが重要です。

BPXPRMxx で定義されるさまざまな限度と、それらが Developer for System z に与える影響の詳細については、227 ページの『第 13 章 チューニングに関する考慮事項』を参照してください。

MAXASSIZE は、最大アドレス・スペース (プロセス) 領域サイズを指定します。SYS1.PARMLIB(BPXPRMxx) で MAXASSIZE を 2G に設定してください。これは、許容される最大値です。これはシステム全体の限度であるため、すべての z/OS UNIX アドレス・スペースに対してアクティブとなります。この値が必要な限度と異なる場合は、ご使用のセキュリティ・ソフトウェアで Developer for System z 独自の限度を設定できます。詳細については、196 ページの『Developer for System z 開始タスクの定義』を参照してください。

MAXTHREADS は、単一プロセスでのアクティブなスレッドの最大数を指定します。SYS1.PARMLIB(BPXPRMxx) で MAXTHREADS を 1500 以上に設定してください。これはシステム全体の限度であるため、すべての z/OS UNIX アドレス・スペースに対してアクティブとなります。この値が必要な限度と異なる場合は、ご使用のセキュリティ・ソフトウェアで Developer for System z 独自の限度を設定できます。詳細については、196 ページの『Developer for System z 開始タスクの定義』を参照してください。

MAXTHREADTASKS は、単一プロセスでのアクティブな MVS タスクの最大数を指定します。SYS1.PARMLIB(BPXPRMxx) で MAXTHREADTASKS を 1500 以上に設定してください。これはシステム全体の限度であるため、すべての z/OS UNIX アドレス・スペースに対してアクティブとなります。この値が必要な限度と異なる場合は、ご使用のセキュリティ・ソフトウェアで Developer for System z 独自の限度を設定できます。詳細については、196 ページの『Developer for System z 開始タスクの定義』を参照してください。

MAXPROCUSER は、単一の z/OS UNIX ユーザー ID が同時にアクティブにしておくことができるプロセスの最大数を指定します。SYS1.PARMLIB(BPXPRMxx) で MAXPROCUSER を 50 以上に設定してください。この設定は、Developer for System z を使用する各クライアントに対してアクティブにする必要があるため、システム全体の限度となります。

これらの値は、以下のコンソール・コマンドで検査し、動的に (次回の IPL まで) 設定できます。

- DISPLAY OMVS,0
- SETOMVS MAXASSIZE=2G
- SETOMVS MAXTHREADS=1500
- SETOMVS MAXTHREADTASKS=1500
- SETOMVS MAXPROCUSER=50

注:

1. アドレス・スペース・サイズを設定または制限できる、その他のロケーションについて詳しくは、162 ページの『アドレス・スペース・サイズ』を参照してください。
2. 上で使用した MAXPROCUSER 値は、固有の z/OS UNIX ユーザー ID (UID) を持つユーザーに基づいています。ユーザーが同じ UID を共用する場合は、この値を大きくしてください。
3. その他の BPXPRMxx 値 (MAXPROCSYS や MAXUIDS などの値) を、同時にアクティブになることが予想される Developer for System z ユーザー数を処理するのに十分な値にしてください。詳細については、227 ページの『第 13 章 チューニングに関する考慮事項』を参照してください。

COMMNDxx への開始タスクの追加

Developer for System z RSED、LOCKD、および JMON サーバーの始動コマンドを SYS1.PARMLIB(COMMNDxx) に追加し、これらのサーバーが次のシステム IPL で自動的に始動するようにします。

サーバーを定義および構成した後、これらのサーバーを以下のコンソール・コマンドで動的に (次の IPL まで) 始動できます。

- S RSED
- S LOCKD
- S JMON

注: ロック・デーモンは、Developer for System z ユーザーが RSE デーモンにログオンする前に始動してください。これは、ロック・デーモンがそれらのユーザーによるデータ・セット・ロック要求を追跡できるようにするためです。したがって、ロック・デーモンはシステムの始動時に始動してください。

LPALSTxx での LPA 定義

(オプションの) 共通アクセス・リポジトリ・マネージャー (CARMA) は、JES イニシエーターを使用せずに済む代替のサーバー始動方式をサポートしています。これらの代替方式で最も柔軟なものは、FEK.SFEKLPA ロード・ライブラリー内のモジュール CRASTART がリンク・バック域 (LPA) 内にあることを必要とします。

LPA データ・セットは、SYS1.PARMLIB(LPALSTxx) で定義されます。

以下のコンソール・コマンドで、LPA 定義を動的に (次の IPL まで) 設定できます。

- SETPROG LPA,ADD,DSN=FEK.SFEKLPA

PROGxx での APF 許可

JES ジョブ・モニターで JES スプール・ファイルにアクセスするためには、FEK.SFEKAUTH ロード・ライブラリー内のモジュール FEJMON と、Language Environment® (LE) ランタイム・ライブラリー (CEE.SCEERUN*) に、APF 許可がある必要があります。

(オプションの) SCLM Developer Toolkit サービスを機能させるためには、FEK.SFEKAUTH ロード・ライブラリー内のモジュール BWBTSOW と REXX ランタイム・ライブラリー (REXX.*.SEAGLPA) に、APF 許可がある必要があります。

ISPF で TSO/ISPF クライアント・ゲートウェイを作成するには、SYS1.LINKLIB 内のモジュール ISPZTSO に APF 許可がある必要があります。TSO/ISPF クライアント・ゲートウェイは、Developer for System z の TSO コマンド・サービス、SCLM Developer Toolkit、およびオプションとして CARMA によって使用されます。

使用しているサイトが IBM の推奨に従っている場合、APF 許可は、SYS1.PARMLIB(PROGxx) の中で定義されています。

APF 許可は、以下のコンソール・コマンドで動的に (次の IPL まで) 設定できます。ここで、volser はデータ・セットが存在するボリューム (SMS の管理対象でない場合) です。

- SETPROG APF,ADD,DSN=FEK.SFEKAUTH,SMS
- SETPROG APF,ADD,DSN=CEE.SCEERUN,VOL=volser
- SETPROG APF,ADD,DSN=CEE.SCEERUN2,VOL=volser
- SETPROG APF,ADD,DSN=REXX.V1R4M0.SEAGLPA,VOL=volser
- SETPROG APF,ADD,DSN=SYS1.LINKLIB,VOL=volser

注:

1. REXX 製品パッケージに代替ライブラリーを使用している場合、デフォルトの REXX ランタイム・ライブラリー名は REXX.*.SEAGALT であり、前のサンプルで使用した REXX.*.SEAGLPA ではありません。
2. REXX.*.SEAGLPA などの LPA ライブラリーは、LPA 内に置かれている場合は自動的に APF 許可があり、したがって明示的な定義は必要ありません。
3. 一部の相互前提条件の製品 (IBM Debug Tool など) も APF 許可を必要とします。これについての詳細は、関連する製品のカスタマイズ・ガイドを参照してください。

PROGxx での LINKLIST 定義

Developer for System z の LINKLIST 定義は、3 つのカテゴリに分けることができます。

- Developer for System z の各機能に必要な Developer for System z ロード・ライブラリー。これらの定義については、このセクションで説明します。
- Developer for System z の各機能に必要なロード・ライブラリー。これらの定義については、22 ページの『必要な LINKLIST 定義と LPA 定義』で説明します。
- 他の製品が必要とする Developer for System z ロード・ライブラリー。これらの定義については、23 ページの『他の製品用の LINKLIST 定義』で説明します。

(オプションの) SCLM Developer Toolkit サービスを機能させるためには、FEK.SFEKAUTH および FEK.SFEKLOAD ロード・ライブラリー内のすべての BWB* モジュールを、STEPLIB または LINKLIST によって使用可能にする必要があります。

STEPLIB を使用する場合は、LINKLIST によって使用できないライブラリーを、rsed.envvars (RSE 構成ファイル) の STEPLIB ディレクティブで定義する必要があります。ただし、次の点に注意してください。

- STEPLIB を z/OS UNIX で使用すると、パフォーマンスに悪い影響が出ます。
- 1 つの STEPLIB ライブラリーに APF 許可がある場合、すべての STEPLIB ライブラリーに APF 許可があることが必要です。ライブラリーは、STEPLIB 内で許可のないライブラリーと混用した場合、APF 許可を失います。

使用しているサイトが IBM の推奨に従っている場合、LINKLIST データ・セットは、SYS1.PARMLIB(PROGxx) の中で定義されています。

必要な定義は以下のようになります。ここで、listname はアクティブにされる LINKLIST セットの名前で、volser はマスター・カタログにカタログされていないデータ・セットが存在しているボリュームです。

- LNKST ADD NAME(listname) DSN=FEK.SFEKAUTH) VOLUME(volser)
- LNKST ADD NAME(listname) DSN=FEK.SFEKLOAD)

LINKLIST 定義は、以下のコンソール・コマンド・グループを使用して動的に (次の IPL まで) 作成できます。ここで、listname は現行 LINKLIST セットの名前で、volser はマスター・カタログにカタログされていないデータ・セットが存在しているボリュームです。

1. LNKST DEFINE,NAME=LLTMP,COPYFROM=CURRENT
2. LNKST ADD NAME=LLTMP,DSN=FEK.SFEKAUTH,VOL=volser
3. LNKST ADD NAME=LLTMP,DSN=FEK.SFEKLOAD
4. LNKST ACTIVATE,NAME=LLTMP
5. LNKST UNDEFINE,NAME=listname
6. LNKST UPDATE,JOB=*

必要な LINKLIST 定義と LPA 定義

リモート・システム・エクスプローラー (RSE) は、MVS ロード・ライブラリーへのアクセスを必要とする z/OS UNIX プロセスです。以下の (前提条件の) ライブラリーは、STEPLIB または LINKLIST/LPALIB によって使用可能であることが必要です。

- システム・ロード・ライブラリー
 - SYS1.LINKLIB
- 言語環境プログラム・ランタイム
 - CEE.SCEERUN
 - CEE.SCEERUN2
- C++ の DLL クラス・ライブラリー
 - CBC.SCLBDLL
- ISPF の TSO/ISPF クライアント・ゲートウェイ

- ISP.SISPLD
- ISP.SISPLPA

オプションのサービスを使用できるようにするには、以下の追加ライブラリーを STEPLIB または LINKLIST/LPALIB を通じて使用可能にする必要があります。このリストには、Developer for System z が対話する製品 (IBM Debug Tool など) に固有のデータ・セットは含まれていません。

- REXX ランタイム・ライブラリー (SCLM Developer Toolkit 用)
 - REXX.*.SEAGLPA
- システム・ロード・ライブラリー (SSL 暗号化用)
 - SYS1.SIEALNKE
- TCP/IP ロード・ライブラリー (TSO コマンド・サービスに APPC を使用している場合)
 - TCPIP.SEZALOAD

注:

1. REXX 製品パッケージに代替ライブラリーを使用している場合、デフォルトの REXX ランタイム・ライブラリー名は REXX.*.SEAGALT であり、前のサンプルで使用した REXX.*.SEAGLPA ではありません。
2. LPA 配置用に設計されたライブラリー (REXX.*.SEAGLPA など) は、LINKLIST または STEPLIB によってアクセスされる場合、追加のプログラム制御か APF 許可、またはその両方を必要とすることがあります。
3. 一部の相互前提条件の製品 (IBM Debug Tool など) も STEPLIB 定義または LINKLIST/LPALIB 定義を必要とします。これについての詳細は、関連する製品のカスタマイズ・ガイドを参照してください。
4. CEE.SCEELKED が LINKLIST または STEPLIB 内にある場合は、TCPIP.SEZALOAD を CEE.SCEELKED の前に配置する必要があります。そうしないと、TCP/IP REXX ソケット呼び出しで OC1 システム異常終了が発生します。

使用しているサイトが IBM の推奨に従っている場合、LINKLIST データ・セットは、SYS1.PARMLIB(PROGxx) の中で定義されています。LPA データ・セットは、SYS1.PARMLIB(LPALSTxx) で定義されます。

STEPLIB を使用する場合は、LINKLIST/LPALIB によって使用できないライブラリーを、rsed.envvars (RSE 構成ファイル) の STEPLIB ディレクティブで定義する必要があります。ただし、次の点に注意してください。

- STEPLIB を z/OS UNIX で使用すると、パフォーマンスに悪い影響が出ます。
- 1 つの STEPLIB ライブラリーに APF 許可がある場合、すべての STEPLIB ライブラリーに APF 許可があることが必要です。ライブラリーは、STEPLIB 内で許可のないライブラリーと混用した場合、APF 許可を失います。
- JCL で STEPLIB DD に追加されたライブラリーは、その JCL によって開始された z/OS UNIX プロセスに伝搬されません。

他の製品用の LINKLIST 定義

Developer for System z クライアントには、エンタープライズ・サービス・ツール (EST) と呼ばれるコード生成コンポーネントがあります。生成コードが診断用エラ

ー・メッセージを発行するためには、FEK.SFEKLOAD ロード・ライブラリー内のすべての IRZ* モジュールと IIRZ* モジュールを、STEPLIB または LINKLIST を通じて使用可能にする必要があります。

使用しているサイトが IBM の推奨に従っている場合、LINKLIST データ・セットは、SYS1.PARMLIB(PROGxx) の中で定義されています。

STEPLIB を使用する場合は、LINKLIST によって使用できないライブラリーを、コード (IMS™ またはバッチ・ジョブ) を実行するタスクの STEPLIB ディレクティブで定義する必要があります。ただし、以下の点に留意してください。

- 1 つの STEPLIB ライブラリーに APF 許可がある場合、すべての STEPLIB ライブラリーに APF 許可があることが必要です。ライブラリーは、STEPLIB 内で許可のないライブラリーと混用した場合、APF 許可を失います。

PROCLIB の変更

以下に示す開始タスク・プロシージャーおよびリモート・ビルド・プロシージャーは、使用する JES サブシステムに対して定義されたシステム・プロシージャー・ライブラリー内に存在する必要があります。以下の説明では、IBM のデフォルトのプロシージャー・ライブラリー SYS1.PROCLIB が使用されています。

JES ジョブ・モニター

サンプルの開始タスク・メンバー FEK.#CUST.PROCLIB(JMON) を、このメンバー内で説明されているようにカスタマイズし、SYS1.PROCLIB にコピーしてください。下記のコード・サンプルに示すように、以下のものを提供する必要があります。

- (許可された) ロード・ライブラリー (デフォルトでは FEK) の高位修飾子
- JES ジョブ・モニター構成ファイル。デフォルトでは FEK.#CUST.PARMLIB (FEJJCNFG)

```
/*
/* JES JOB MONITOR
/*
/*JMON      PROC PRM=,          * PRM='-TV' TO START TRACING
/*          LEPRM='RPTOPTS(ON)',
/*          HLQ=FEK,
/*          CFG=FEK.#CUST.PARMLIB(FEJJCNFG)
/*
/*JMON      EXEC PGM=FEJJMON,REGION=0M,TIME=NOLIMIT,
/*          PARM=('&LEPRM,ENVAR("&CEE_ENVFILE_S=DD:ENVIRON")/&PRM')
/*STEPLIB DD DISP=SHR,DSN=&HLQ..SF&EKAUTH
/*ENVIRON DD DISP=SHR,DSN=&CFG
/*SYSPRINT DD SYSOUT=*
/*SYSOUT DD SYSOUT=*
/*          PEND
/*
```

図 1. JMON - JES ジョブ・モニター開始タスク

注:

1. 始動パラメーターの詳細については、133 ページの『第 8 章 オペレーター・コマンド』を参照してください。

2. サンプル JCL は、初期には FEK.SFEKSAMP(FEJJJCL) として出荷され、18 ページの『カスタマイズのセットアップ』で FEK.#CUST.PROCLIB(JMON) に名前変更されました。
3. トレースは、133 ページの『第 8 章 オペレーター・コマンド』で説明されているように、コンソール・コマンドによっても制御できます。
4. このタスクは、ワークロード・マネージャー (WLM) で SYSSTC または同等の目標に割り当てる必要があります。
5. LE 環境変数 _CEE_ENVFILE_S を使用するには、z/OS 1.8 以上が必要です。これより前の z/OS レベルでは、この変数を _CEE_ENVFILE で置き換えることができますが、C ランタイムのバグにより、JES ジョブ・モニター構成ファイル (FEJJCNFG) 内の TZ 変数が正しく解釈されない場合があります。

RSE デーモン

サンプルの開始タスク・メンバー FEK.#CUST.PROCLIB(RSED) を、このメンバー内で説明されているようにカスタマイズし、SYS1.PROCLIB にコピーしてください。下記のコード・サンプルに示すように、以下のものを提供する必要があります。

- RSE デーモン・ポート。デフォルトでは 4035。
- Developer for System z がインストールされているホーム・ディレクトリー、デフォルトは /usr/lpp/rdz。
- 構成ファイルのロケーション。デフォルトでは /etc/rdz

```

/*
/* RSE DAEMON
/*
//RSED      PROC IVP='',                * 'IVP' to do an IVP test
//          PORT=4035,
//          HOME='/usr/lpp/rdz',
//          CNFG='/etc/rdz'
/*
//RSE       EXEC PGM=BPXBATSL,REGION=0M,TIME=NOLIMIT,
//          PARM='PGM &HOME/bin/rsed.sh &IVP &PORT &CNFG'
//STDERR    DD SYSOUT=*
//STDOUT    DD SYSOUT=*
//          PEND
/*

```

図2. RSED - RSE デーモン開始タスク

注:

- 始動パラメーターの詳細については、133 ページの『第 8 章 オペレーター・コマンド』を参照してください。
- サンプル JCL は、最初に FEK.SFEKSAMP(FEKRSED) として出荷され、18 ページの『カスタマイズのセットアップ』で FEK.#CUST.PROCLIB(RSED) に名前変更されます。
- ジョブ名の長さは、7 文字かそれ以下に制限してください。8 文字の名前を使用した場合、**modify** および **stop** オペレーター・コマンドは、「IEE342I MODIFY REJECTED-TASK BUSY」というメッセージとともに失敗します。この動作は、z/OS UNIX の子プロセスの設計によるものです。
- このタスクおよびこのタスクが作成する子プロセスは、ワークロード・マネージャー (WLM) で SYSSTC または同等の目標に割り当てる必要があります。

す。子プロセスには、親タスクの名前 RSED にランダムな 1 桁の数字を付加した名前が付けられます (RSED8 など)。

ロック・デーモン

サンプルの開始タスク・メンバー FEK.#CUST.PROCLIB(LOCKD) を、このメンバー内で説明されているようにカスタマイズし、SYS1.PROCLIB にコピーしてください。下記のコード・サンプルに示すように、以下のものを提供する必要があります。

- Developer for System z がインストールされているホーム・ディレクトリー。デフォルトでは /usr/lpp/rdz。
- 構成ファイルのロケーション。デフォルトでは /etc/rdz。
- 初期のログ詳細レベル。デフォルトでは 1。

```
/*  
/* RSE LOCK DAEMON  
/*  
//LOCKD   PROC HOME='/usr/lpp/rdz',  
//         CNFG='/etc/rdz',  
//         LOG=1  
/*  
//LOCKD   EXEC PGM=BPXBATSL,REGION=0M,TIME=NOLIMIT,  
//         PARM=PGM &HOME./bin/lockd.sh &CNFG &LOG'  
//STDOUT   DD SYSOUT=*  
//STDERR   DD SYSOUT=*  
//         PEND  
/*
```

図 3. LOCKD - ロック・デーモン開始タスク

注:

1. 始動パラメーターの詳細については、133 ページの『第 8 章 オペレーター・コマンド』を参照してください。
2. サンプル JCL は、初期には FEK.SFEKSAMP(FEKLCKD) として出荷され、18 ページの『カスタマイズのセットアップ』で FEK.#CUST.PROCLIB(LOCKD) に名前変更されました。
3. このタスクは、ワークロード・マネージャー (WLM) で SYSSTC または同等の目標に割り当てる必要があります。

PARM 変数の JCL での制限

PARM 変数の最大長は 100 文字であり、これが原因でカスタム・ディレクトリー名を使用する場合に問題が起きることもあります。この問題を回避するには、以下のいずれかを行います。

- シンボリック・リンクを使用する

長いディレクトリー名の省略方法として、シンボリック・リンクを使用できます。次のサンプルの z/OS UNIX コマンドは、別のディレクトリー (/long/directory/name/usr/lpp/rdz) へのシンボリック・リンク (/usr/lpp/rdz) を定義します。

```
ln -s /long/directory/name/usr/lpp/rdz /usr/lpp/rdz
```

- STDIN を使用する

PARM フィールドが空の場合、BPXBATSL は z/OS UNIX シェルを始動し、STDIN から提供されたシェル・スクリプトを実行します。STDIN は z/OS UNIX ファイル (ORDONLY として割り振られたもの) でなければならず、STDIN を使用すると、ポートなどに PROC 変数を使用できなくなります。また、シェルがシェル・ログオン・スクリプト /etc/profile および \$HOME/.profile を実行することにも注意してください。

この方法を使用するには、最初に始動 JCL を更新して、以下のサンプルのようなものに一致させる必要があります。

```

/*
/* RSE DAEMON - USING STDIN
/*
//RSED      PROC CNFG='/etc/rdz'
/*
//RSE       EXEC PGM=BPXBATSL,REGION=0M,TIME=NOLIMIT
//STDOUT    DD SYSOUT=*
//STDERR    DD SYSOUT=*
//STDIN     DD PATHOPTS=(ORDONLY),PATH='&CNFG./rsed.stdin.sh'
//STDENV    DD PATHOPTS=(ORDONLY),PATH='&CNFG./rsed.envvars'
//          PEND
/*

```

図 4. RSED - 代替 RSE デーモンの始動

次に、RSE デーモンを始動するシェル・スクリプト (この例では /etc/rdz/rsed.stdin.sh) を作成する必要があります。このスクリプトの内容は、次のサンプルのようになります。

```

/long/directory/name/usr/lpp/rdz/bin/rsed.sh 4035 /etc/rdz

```

図 5. rsed.stdin.sh - 代替 RSE デーモンの始動

注: この始動方式を使用する場合は、デーモン始動 JCL で STDENV に rsed.envvars を割り振ってください。rsed.envvars には、システム・リソースの節約に役立ついくつかの z/OS UNIX ディレクティブが定義されています。

ELAXF* リモート・ビルド・プロシージャ

Developer for System z は、CICS BMS マップ、IMS MFS 画面、および COBOL、PL/I、アセンブラー、C/C++ の各プログラムの、JCL 生成、リモート・プロジェクト・ビルド、およびリモート構文検査の各フィーチャーに使用できるサンプル JCL プロシージャを提供します。これらのプロシージャを使用すると、インストールごとに独自の標準を適用でき、開発者は、同じプロシージャを同じコンパイラー・オプションおよびコンパイラー・レベルで使用できます。

サンプル・プロシージャとその機能を、表 7 に示します。

表 7. サンプル ELAXF* プロシージャ

メンバー	目的
ELAXFADT	高水準アセンブラー・プログラムのアセンブルとデバッグのためのサンプル・プロシージャ。

表 7. サンプル ELAXF* プロシージャ (続き)

メンバー	目的
ELAXFASM	高水準アセンブラー・プログラムのアセンブルのためのサンプル・プロシージャ。
ELAXFBMS	CICS BMS オブジェクトおよびそれに対応する copy、dsect、または include メンバーを作成するためのサンプル・プロシージャ。
ELAXFCOC	COBOL コンパイル、統合 CICS 変換、および統合 DB2 変換を行うためのサンプル・プロシージャ。
ELAXFCOP	COBOL プログラムに埋め込まれた EXEC SQL ステートメントの DB2 プリプロセスを行うためのサンプル・プロシージャ。
ELAXFCOT	COBOL プログラムに埋め込まれた EXEC CICS ステートメントの CICS 変換を行うためのサンプル・プロシージャ。
ELAXFCPC	C コンパイルを行うためのサンプル・プロシージャ。
ELAXFCPP	C++ コンパイルを行うためのサンプル・プロシージャ。
ELAXFCPI	SCM プリプロセッサ・ステートメント (-INC および ++INCLUDE) を使用した COBOL コンパイルのためのサンプル・プロシージャ。
ELAXFDCL	プログラムを TSO モードで実行するためのサンプル・プロシージャ。
ELAXFGO	GO ステップのためのサンプル・プロシージャ。
ELAXFLNK	C/C++、COBOL、PLI、および高水準アセンブラーの各プログラムをリンクするためのサンプル・プロシージャ。
ELAXFMFS	IMS MFS 画面を作成するためのサンプル・プロシージャ。
ELAXFPLP	PLI プログラムに埋め込まれた EXEC SQL ステートメントの DB2 プリプロセスを行うためのサンプル・プロシージャ。
ELAXFPLT	PLI プログラムに埋め込まれた EXEC CICS ステートメントの CICS 変換を行うためのサンプル・プロシージャ。
ELAXFPLI	PL/I コンパイル、統合 CICS 変換、および統合 DB2 変換を行うためのサンプル・プロシージャ。
ELAXFPPI	SCM プリプロセッサ・ステートメント (-INC および ++INCLUDE) を使用した PL/I コンパイルのためのサンプル・プロシージャ。
ELAXFTSO	生成された DB2 コードを TSO モードで実行/デバッグするためのサンプル・プロシージャ。
ELAXFUOP	CICS または IMS サブシステムで実行するプログラムをビルドするときに、UOPT ステップを生成するためのサンプル・プロシージャ。

プロシージャの名前とプロシージャ内のステップの名前は、Developer for System z クライアントの出荷時に添付されるデフォルトのプロパティに一致しています。プロシージャの名前またはプロシージャ内のステップの名前を変更する場合は、すべてのクライアント上の対応するプロパティ・ファイルも更新する必要があります。プロシージャ名とステップ名は、変更しないことをお勧めします。

サンプルのビルド・プロシージャ・メンバー FEK.#CUST.PROCLIB(ELAXF*) を、各メンバー内で説明されているようにカスタマイズし、SYS1.PROCLIB にコピーします。さまざまな製品ライブラリーに、29 ページの表 8 に示した正しい高位修飾子を指定する必要があります。

表 8. ELAXF* 高位修飾子チェックリスト

製品	デフォルト HLQ	値
Developer for System z	FEK	
CICS	CICSTS32.CICS	
DB2	DSN910	
IMS	IMS	
COBOL	IGY.V4R1M0	
PL/I	IBMZ.V3R8M0	
C/C++	CBC	
LE	CEE	
システム LINKLIB	SYS1	
システム MACLIB	SYS1	

ELAXF* プロシージャーをシステム・プロシージャー・ライブラリーにコピーできない場合は、クライアント上のジョブ・プロパティーに JCLLIB カードを (JOB カードの直後に) 追加するように、Developer for System z ユーザーに依頼します。

```
//MYJOB    JOB <job parameters>
//PROCS    JCLLIB ORDER=(FEK.#CUST.PROCLIB)
```

セキュリティ定義

データ・セット FEK.#CUST.JCL 内のサンプル・メンバー FEKRACF をカスタマイズおよび実行依頼して、Developer for System z のセキュリティ定義を作成してください。このジョブを実行依頼するユーザーは、RACF SPECIAL などのセキュリティ管理者特権を持っている必要があります。

注:

- CA ACF2™ for z/OS を使用しているサイトの場合は、
<https://support.ca.com/irj/portal/kbtech?ipLogNrow=0&docid=492389&searchID=TEC492389> にアクセスして、Developer for System z を正しく構成するために必要なセキュリティ・コマンドの詳細を参照してください。
- CA Top Secret® for z/OS を使用しているサイトの場合は、CA サポート・サイト (<https://support.ca.com>) で、ご使用の製品のページを参照し、関連する Developer for System z Knowledge Document を確認してください。この Knowledge Document には、Developer for System z を正しく構成するために必要なセキュリティ・コマンドの詳細が記載されています。

以下にリストする Developer for System z の必須のセキュリティ関連定義については、169 ページの『第 10 章 セキュリティーに関する考慮事項』で詳しく説明します。この章では、Developer for System z のセキュリティに関する一般的な側面についても説明しています (サンプルの FEKRACF ジョブでは対応していない必要製品のセキュリティの側面など)。

- セキュリティーの設定およびクラスをアクティブにする
- Developer for System z ユーザーの OMVS セグメントを定義する
- データ・セット・プロファイルを定義する

- JMON、RSED、および LOCKD 開始タスクの定義
- JES コマンド・セキュリティを定義する
- セキュアな z/OS UNIX サーバーとして RSE を定義する
- RSE の MVS プログラム制御ライブラリーを定義する
- RSE のアプリケーション・セキュリティを定義する
- RSE の PassTicket サポートを定義する
- RSE の z/OS UNIX プログラム制御ファイルを定義する

注: サンプルの FEKRACF ジョブは、単なる RACF コマンドを上回る機能を備えています。セキュリティ定義の最後のステップでは、z/OS UNIX ファイルをプログラムで制御されるようにします。使用するサイトのポリシーによっては、これはセキュリティ管理者でなく、システム・プログラマーの作業である場合もあります。

重要: アプリケーション・セキュリティおよび PassTicket が正しくセットアップされていないと、クライアントの接続要求は失敗します。

FEJJCNFG、JES ジョブ・モニター構成ファイル

JES ジョブ・モニター (JMON) は、JES に関連したすべてのサービスを提供します。JES ジョブ・モニターの動作は、FEJJCNFG 内の定義によって制御できます。

FEJJCNFG は FEK.#CUST.PARMLIB に置かれます。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

サンプルの JES ジョブ・モニター構成メンバー、FEJJCNFG を、次の例に示すようにカスタマイズしてください。US コード・ページを使用する場合、コメント行はポンド記号 (#) で始まります。データ行には、ディレクティブとそれに割り当てられる値のみを入れることができ、その同じ行にコメントを入れることはできません。

注: 加えた変更があれば、それを取得するために JMON 開始タスクを再始動する必要があります。

```

SERV_PORT=6715
TZ=EST5EDT
#_BPXK_SETIBMOPT_TRANSPORT=TCPIP
#APPLID=FEKAPPL
#AUTHMETHOD=SAF
#CODEPAGE=UTF-8
#CONCHAR=$
#CONSOLE_NAME=JMON
#GEN_CONSOLE_NAME=OFF
#HOST_CODEPAGE=IBM-1047
#LIMIT_COMMANDS=NOLIMIT
#LIMIT_VIEW=USERID
#LISTEN_QUEUE_LENGTH=5
#MAX_DATASETS=32
#MAX_THREADS=200
#TIMEOUT=3600
#TIMEOUT_INTERVAL=1200
#SUBMITMETHOD=TSO
#TSO_TEMPLATE=FEK.#CUST.CNTL(FEJTSO)

```

図 6. FEJJCNGF、JES ジョブ・モニター構成ファイル

SERV_PORT

JES ジョブ・モニター・ホスト・サーバーのポート番号。デフォルト・ポートは 6715 です。任意の値に変更してもかまいませんが、サーバーと Developer for System z クライアントの両方を同じポート番号で構成する必要があります。サーバー・ポート番号を変更した場合は、すべてのクライアントも、このシステムの JES ジョブ・モニター・ポートを「リモート・システム」ビューで変更する必要があります。

注:

- ポートを選択する前に、ご使用のシステムで、そのポートが TSO コマンド **NETSTAT** および **NETSTAT PORTL** で使用可能であることを確認してください。
- バージョン 7.1 以上のクライアントを使用している場合、このポート上のすべての通信は、ご使用の z/OS ホスト・マシンだけに限定されます。

TZ タイム・ゾーン・セレクター。デフォルトは EST5EDT です。デフォルトのタイム・ゾーンは UTC +5 時間 (米東部標準時 (EST) 夏時間 (EDT)) です。これは、使用するタイム・ゾーンを表すように変更してください。詳細は、「*UNIX System Services コマンド解説書*」(SA88-8641) に記載されています。

以下の定義はオプションです。省略した場合は、以下に示すデフォルト値が使用されます。

_BPXK_SETIBMOPT_TRANSPORT

使用する TCPIP スタックの名前を指定します。デフォルトは TCPIP です。コメント解除して、要求された TCPIP スタック名に変更してください。この名前は、関連する TCPIP.DATA 内の TCPIPJOBNAME ステートメントで定義されています。

注:

- サーバー JCL 内の SYSTCPD DD ステートメントのコーディングでは、要求されたスタックのアフィニティーは設定されません。
- このディレクティブがアクティブでないときは、JES ジョブ・モニターがシステム上の使用可能なすべてのスタックにバインドします (BIND INADDRANY)。

APPLID

セキュリティ・ソフトウェアに対して JES ジョブ・モニターを識別するために使用するアプリケーション ID を指定します。デフォルトは FEKAPPL です。コメント解除し、希望するアプリケーション ID に変更してください。

注: この値は、rsed.envvars 構成ファイル内で RSE に設定したアプリケーション ID に一致する必要があります。これらの値が異なる場合、RSE はクライアントを JES ジョブ・モニターに接続できません。

AUTHMETHOD

デフォルトは SAF で、System Authorization Facility (SAF) セキュリティ・インターフェースが使用されることを意味します。IBM サポートから変更するように指示された場合以外は、これを変更しないでください。

CODEPAGE

ワークステーション・コード・ページ。デフォルトは UTF-8 です。ワークステーション・コード・ページは UTF-8 に設定されており、通常、これは変更しないでください。通貨記号など、NLS 文字に問題があるときは、ディレクティブをコメント解除し、ワークステーションのコード・ページに一致するように UTF-8 の変更が必要な場合があります。

CONCHAR

JES コンソール・コマンド文字を指定します。CONCHAR のデフォルトは、JES2 の場合は CONCHAR=\$、JES3 の場合は CONCHAR=* です。コメント解除し、要求されたコマンド文字に変更してください。

CONSOLE_NAME

ジョブに対するコマンド (「保留」、「保留解除」、「キャンセル」、および「ページ」) を発行するために使用する、EMCS コンソールの名前を指定します。デフォルトは JMON です。コメント解除し、以下のガイドラインを使用して、希望するコンソール名に変更してください。

- CONSOLE_NAME は、2 から 8 文字の英数字のコンソール名か、&SYSUID (引用符は付けない) でなければなりません。
- コンソール名を指定した場合、その名前で単一のコンソールがすべてのユーザーによって使用されます。たまたまその名前ですでにコンソールが使用されている場合、クライアントが発行したコマンドは失敗します。
- &SYSUID を指定した場合は、クライアント・ユーザー ID がコンソール名として使用されます。このため、ユーザーごとに異なるコンソールが使用されます。たまたまその名前ですでにコンソールが使用されている場合 (例えば、ユーザーが SDSF ULOG を使用している場合)、クライアントが発行したコマンドは、GEN_CONSOLE_NAME の設定によっては、失敗する場合があります。

どのようなコンソール名を使用しても、コマンドを要求しているクライアントのユーザー ID がコンソールの LU として使用され、トレースでは syslog メッセージ IEA630I および IEA631I が残されます。

```
IEA630I OPERATOR console NOW ACTIVE,  SYSTEM=sysid, LU=id
IEA631I OPERATOR console NOW INACTIVE, SYSTEM=sysid, LU=id
```

GEN_CONSOLE_NAME

代替コンソール名の自動生成を使用可能または使用不可にします。デフォルトは OFF です。代替コンソール名を使用可能にするには、コメント解除して ON に変更します。

このディレクティブを使用するのは、CONSOLE_NAME が &SYSUID と等しく、かつユーザー ID をコンソール名として使用できない場合に限られます。

GEN_CONSOLE_NAME=ON の場合は、ユーザー ID に数字を 1 つだけ付加することによって、代替コンソール名が生成されます。数字 0 から 9 までが試みられます。使用可能なコンソールがない場合、クライアントが発行したコマンドは失敗します。

GEN_CONSOLE_NAME=OFF の場合、クライアントが発行したコマンドは失敗します。

注: 有効な設定値は、ON と OFF だけです。

HOST_CODEPAGE

ホストのコード・ページ。デフォルトは IBM-1047 です。コメント解除し、ご使用のホスト・コード・ページに合わせて変更してください。

バージョン 7.6.1 より、Developer for System z クライアントは、ここで指定された HOST_CODEPAGE 値を無視し、「MVS Files」サブシステムのプロパティーでローカルに指定されたコード・ページを使用するようになりました。

注: JES ジョブ・モニターは、最新のクライアントに対しても、初期のクライアント通信セットアップで HOST_CODEPAGE に指定されたホスト・コード・ページを使用します。

LIMIT_COMMANDS

ユーザーがどのジョブに対して、選択された JES コマンド（「JCL の表示」、「保留」、「保留解除」、「キャンセル」、および「ページ」）を発行できるかを定義します。デフォルト (LIMIT_COMMANDS=USERID) では、コマンドの発行対象は、そのユーザーが所有するジョブだけに制限されます。ユーザーが、セキュリティ製品による許可があれば、すべてのスプール・ファイルに対してコマンドを発行できるようにするには、このディレクティブをコメント解除し、LIMITED または NOLIMIT を指定します。

表 9. LIMIT_COMMANDS コマンドの許可のマトリックス

	ジョブ所有者	
LIMIT_COMMANDS	ユーザー	その他
USERID (デフォルト)	許可される	許可されない
LIMITED	許可される	セキュリティ・プロファイルによって明示的に許可された場合にのみ許可される

表 9. *LIMIT_COMMANDS* コマンドの許可のマトリックス (続き)

	ジョブ所有者	
NOLIMIT	許可される	セキュリティ・プロファイルによって許可された場合、または JESSPOOL クラスがアクティブでない場合は許可される

注: 有効な設定値は、USERID、LIMITED、および NOLIMIT だけです。

LIMIT_VIEW

ユーザーが表示できる出力を定義します。デフォルト (LIMIT_VIEW=NOLIMIT) では、ユーザーはセキュリティ製品による許可があれば、すべての JES 出力を表示できます。表示をそのユーザーが所有している出力だけに制限するには、このディレクティブをコメント解除し、USERID を指定します。

注: 有効な設定値は、USERID と NOLIMIT だけです。

LISTEN_QUEUE_LENGTH

TCP/IP listen キュー長。デフォルトは 5 です。IBM サポートから変更するように指示された場合以外は、これを変更しないでください。

MAX_DATASETS

JES ジョブ・モニターがクライアント (例えば、SYSOUT、SYSPRINT、SYS00001 など) に返すスプール出力データ・セットの最大数。デフォルトは 32 です。最大値は 2147483647 です。

MAX_THREADS

1 つの JES ジョブ・モニターを一度に使用できるユーザーの最大数。デフォルトは 200 です。最大値は 2147483647 です。この数を大きくすると、JES ジョブ・モニターのアドレス・スペースのサイズを大きくしなければならない場合があります。

TIMEOUT

クライアントとの対話がないためにスレッドが強制終了されるまでの時間の長さ (秒単位)。デフォルトは 3600 (1 時間) です。最大値は 2147483647 です。TIMEOUT=0 は、この機能を無効にします。

TIMEOUT_INTERVAL

タイムアウト検査の間隔を表す秒数。デフォルトは 1200 です。最大値は 2147483647 です。

SUBMITMETHOD=TSO

TSO によるジョブの実行依頼。デフォルト (SUBMITMETHOD=JES) では、ジョブが JES へ直接実行依頼されます。TSO の **SUBMIT** コマンドによってジョブを実行依頼するには、このディレクティブをコメント解除して TSO を指定します。この方法を使用すると、TSO 出口を呼び出すことができます。しかし、パフォーマンスが低下するため、お勧めできません。

注:

- 有効な設定値は、TSO と JES だけです。

- SUBMITMETHOD=TSO を指定した場合は、TSO_TEMPLATE も定義する必要があります。

TSO_TEMPLATE

TSO によってジョブを実行依頼するためのラッパー JCL。デフォルト値は FEK.#CUST.CNTL(FEJTSO) です。このステートメントは、TSO 実行依頼のラッパーとして使用される JCL の完全修飾メンバー名を参照します。詳細については、SUBMITMETHOD ステートメントを参照してください。

注:

- サンプルのラッパー・ジョブは FEK.#CUST.CNTL(FEJTSO) で提供されています。必要なカスタマイズの詳細については、このメンバーを参照してください。
- TSO_TEMPLATE は、SUBMITMETHOD=TSO も指定されていないと効果がありません。

rsed.envvars、RSE 構成ファイル

RSE ロック・デーモンおよび RSE サーバー・プロセス (RSE デーモン、RSE スレッド・プール、および RSE サーバー) は、rsed.envvars 内の定義を使用します。オプションの Developer for System z サービスおよびサード・パーティー・サービスも、この構成ファイルを使用して、使用する環境変数を定義することができます。

リモート・システム・エクスプローラー (RSE) は、クライアントをホストに接続したり、特定のサービス用に他のサーバーを始動するなどの、コア・サービスを提供します。ロック・デーモンは、データ・セット・ロックのトラッキング・サービスを提供します。

rsed.envvars は /etc/rdz/ に置かれます。ただし、ジョブ FEK.SFEKSAMP (FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。このファイルは、TSO OEDIT コマンドで編集できます。

以下のサンプルの rsed.envvars ファイルを参照してください。このファイルは、使用するシステム環境に合わせてカスタマイズする必要があります。US コード・ページを使用する場合、コメント行はポンド記号 (#) で始まります。データ行には、ディレクティブとそれに割り当てられる値のみを入れることができ、その同じ行にコメントを入れることはできません。行の継続および等号 (=) の前後のスペースはサポートされていません。

注: 加えた変更があれば、それを取得するために RSED 開始タスクと LOCKD 開始タスクを再始動する必要があります。


```

#=====
# (1) required definitions
JAVA_HOME=/usr/lpp/java/J5.0
RSE_HOME=/usr/lpp/rdz
_RSE_LOCKD_PORT=4036
_RSE_HOST_CODEPAGE=IBM-1047
TZ=EST5EDT
LANG=C
PATH=/bin:/usr/sbin
_CEE_DMPTARG=/tmp
STEPLIB=NONE
#STEPLIB=$STEPLIB:CEE.SCEERUN:CEE.SCEERUN2:CBC.SCLBDLL
_RSE_SAF_CLASS=/usr/include/java_classes/IRRRacf.jar
_RSE_JAVAOPTS=""
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Xms1m -Xmx256m"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Ddaemon.log=/var/rdz/logs"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Duser.log=/var/rdz/logs"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_LOG_DIRECTORY="
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dmaximum.clients=60"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dmaximum.threads=1000"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dminimum.threadpool.process=1"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dmaximum.threadpool.process=100"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dipv6=true"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dkeep.last.log=true"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Denable.standard.log=true"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Denable.port.of.entry=true"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Denable.certificate.mapping=false"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Denable.automount=true"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Denable.audit.log=true"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Daudit.cycle=30"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Daudit.retention.period=0"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Ddeny.nonzero.port=true"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dsingle.logon=false"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dprocess.cleanup.interval=0"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DAPPLID=FEKAPPL"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDENY_PASSWORD_SAVE=true"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dhide_zos_unix=true"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_IDLE_SHUTDOWN_TIMEOUT=3600000"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_TRACING_ON=true"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_MEMLOGGING_ON=true"
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DTSO_SERVER=APPC"
#=====
# (2) required definitions for TSO/ISPF Client Gateway
_CMDSERV_BASE_HOME=/usr/lpp/ispf
_CMDSERV_CONF_HOME=/etc/rdz
_CMDSERV_WORK_HOME=/var/rdz
#STEPLIB=$STEPLIB:ISP.SISPLPA:SYS1.LINKLIB
_RSE_CMDSERV_OPTS=""
_RSE_CMDSERV_OPTS="$_RSE_CMDSERV_OPTS&ISPPROF=&SYSUID..ISPPROF"
#=====
# (3) required definitions for SCLM Developer Toolkit
_SCLMDT_CONF_HOME=/var/rdz/sclmdt
#STEPLIB=$STEPLIB:FEK.SFEKAUTH:FEK.SFEKLOAD
#_SCLMDT_TRANTABLE=FEK.#CUST.LSTRANS.FILE
_ANT_HOME=/usr/lpp/Apache/Ant/apache-ant-1.7.1
#=====
# (4) optional definitions
#_RSE_PORTRANGE=8108-8118
#_BPXK_SETIBMOPT_TRANSPORT=TCPIP
#_FEKFSCMD_TP_NAME=FEKFRSRV
#_FEKFSCMD_PARTNER_LU=lu_name
#GSK_CRL_SECURITY_LEVEL=HIGH
#GSK_LDAP_SERVER=ldap_server_url
#GSK_LDAP_PORT=ldap_server_port
#GSK_LDAP_USER=ldap_userid
#GSK_LDAP_PASSWORD=ldap_server_password
#=====

```

```

# (5) do not change unless directed by IBM support center
_CEE_RUNOPTS="ALL31(ON) HEAP(32M,32K,ANYWHERE,KEEP,,) TRAP(ON)"
_BPX_SHAREAS=YES
_BPX_SPAWN_SCRIPT=YES
JAVA_PROPAGATE=NO
RSE_LIB=$RSE_HOME/lib
PATH=.:$JAVA_HOME/bin:$RSE_HOME/bin:$CMDSESV_BASE_HOME/bin:$PATH
LIBPATH=$JAVA_HOME/bin:$JAVA_HOME/bin/classic:$RSE_LIB:$RSE_LIB/icuc
LIBPATH=.:usr/lib:$LIBPATH
CLASSPATH=$RSE_LIB:$RSE_LIB/dstore_core.jar:$RSE_LIB/clientserver.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/dstore_extra_server.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/zosserver.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/dstore_miners.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/universalminers.jar:$RSE_LIB/mvsminers.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/carma.jar:$RSE_LIB/luceneminer.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/mvsluceneminer.jar:$RSE_LIB/cdzminer.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/mvscdzminer.jar:$RSE_LIB/jesminers.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/FAMiner.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/mvsutil.jar:$RSE_LIB/jesutils.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/lucene-core-2.3.2.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/cdtparser.jar
CLASSPATH=$CLASSPATH:$RSE_LIB/wdzbidi.jar:$RSE_LIB/fmiExtensions.jar
CLASSPATH=$CLASSPATH:$RSE_SAF_CLASS
CLASSPATH=.:$CLASSPATH
_RSE_CMDSESV_OPTS="&SESSION=SPAWN$ RSE_CMDSESV_OPTS"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DISPF_OPTS='$_RSE_CMDSESV_OPTS'"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DA_PLUGIN_PATH=$RSE_LIB"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Xbootclasspath/p:$RSE_LIB/bidiTools.jar"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dfile.encoding=$_RSE_HOST_CODEPAGE"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dconsole.encoding=$_RSE_HOST_CODEPAGE"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_SPIRIT_ON=true"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DSPIRIT_EXPIRY_TIME=6"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DSPIRIT_INTERVAL_TIME=6"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dcom.ibm.cacheLocalHost=true"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Duser.home=$HOME"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dclient.username=$RSE_USER_ID"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dlow.heap.usage.ratio=15"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dmaximum.heap.usage.ratio=40"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_KEEPALIVE_ENABLED=true"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_KEEPALIVE_RESPONSE_TIMEOUT=30000"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DDSTORE_IO_SOCKET_READ_TIMEOUT=90000"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DRSECOMM_LOGFILE_MAX=0"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dlock.daemon.port=$_RSE_LOCKD_PORT"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dlock.daemon.cleanup.interval=1440"
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -showversion"
_RSE_SERVER_CLASS=org.eclipse.dstore.core.server.Server
_RSE_DAEMON_CLASS=com.ibm.etools.zos.server.RseDaemon
_RSE_POOL_SERVER_CLASS=com.ibm.etools.zos.server.ThreadPoolProcess
_RSE_LOCKD_CLASS=com.ibm.ftt.rse.mvs.server.miners.MVSLockDaemon
_RSE_SERVER_TIMEOUT=120000
_SCLMDT_BASE_HOME=$RSE_HOME
_SCLMDT_WORK_HOME=$CMDSESV_WORK_HOME
CGI_DTWORK=$_SCLMDT_WORK_HOME
#=====
# (6) additional environment variables

```

図 8. (続き)

注: `rsed.envvars` でディレクトリーを指定するときに、シンボリック・リンクを使用できます。

以下の定義が必要です。

JAVA_HOME

Java ホーム・ディレクトリー。デフォルトは `/usr/lpp/java/J5.0` です。使用する Java インストール済み環境に合わせて変更してください。

RSE_HOME

RSE ホーム・ディレクトリー。デフォルトは `/usr/lpp/rdz` です。使用する Developer for System z インストール済み環境に合わせて変更してください。

_RSE_LOCKD_PORT

RSE ロック・デーモン・ポート番号。デフォルトは 4036 です。必要であれば変更できます。

注:

- ポートを選択する前に、ご使用のシステムで、そのポートが TSO コマンド **NETSTAT** および **NETSTAT PORTL** で使用可能であることを確認してください。
- このポート上のすべての通信は、ご使用の z/OS ホスト・マシンだけに限定されます。

_RSE_HOST_CODEPAGE

ホストのコード・ページ。デフォルトは IBM-1047 です。使用するホスト・コード・ページに合わせて変更してください。

TZ タイム・ゾーン・セレクター。デフォルトは EST5EDT です。デフォルトのタイム・ゾーンは UTC +5 時間 (米東部標準時 (EST) 夏時間 (EDT)) です。使用するタイム・ゾーンに合わせて変更してください。

詳細は、「*UNIX System Services* コマンド解説書」(SA88-8641) に記載されています。

LANG デフォルト・ロケールの名前を指定します。デフォルトは C です。C は POSIX ロケールを指定し、(例えば) Ja_JP は日本語ロケールを指定します。使用するロケールに合わせて変更してください。

PATH コマンド・パス。デフォルトは `/bin:/usr/sbin:.` です。必要であれば変更できます。

_CEE_DMPTARG

Java 仮想マシン (JVM) によって使用される、言語環境プログラム (LE) z/OS UNIX ダンプ・ロケーション。デフォルトは `/tmp` です。

STEPLIB

LINKLIST/LPALIB でなく、MVS データ・セットにアクセスします。デフォルトは NONE です。

以下の 1 つ以上の STEPLIB ディレクティブをコメント解除してカスタマイズすることにより、(前提条件の) ライブラリーを LINKLIST/LPALIB 内

に保持せずに済むことができます。以下にリストしたライブラリーの詳しい使用方法については、19 ページの『PARMLIB の変更』を参照してください。

```
STEPLIB=$STEPLIB:CEE.SCEERUN:CEE.SCEERUN2:CBC.SCLBDLL
STEPLIB=$STEPLIB:ISP.SISPLD:ISP.SISPLPA:SYS1.LINKLIB
STEPLIB=$STEPLIB:FEK.SFEKAUTH:FEK.SFEKLOAD
```

注:

- STEPLIB を z/OS UNIX で使用すると、パフォーマンスに悪い影響が出ます。
- 1 つの STEPLIB ライブラリーに APF 許可がある場合、すべての STEPLIB ライブラリーに APF 許可があることが必要です。ライブラリーは、STEPLIB 内で許可のないライブラリーと混用した場合、APF 許可を失います。
- LPA 配置用に設計されたライブラリーは、LINKLIST または STEPLIB によってアクセスされる場合に、追加のプログラム制御および APF 許可を必要とすることがあります。
- サーバー JCL 内の STEPLIB DD ステートメントのコーディングでは、要求された STEPLIB 連結は設定されません。

RSE_SAF_CLASS

セキュリティー製品への Java インターフェースを指定します。デフォルトは /usr/include/java_classes/IRRRacf.jar です。ご使用のセキュリティー・ソフトウェアのセットアップに合わせて変更してください。

注: z/OS 1.10 以降、/usr/include/java_classes/IRRRacf.jar はベース z/OS に添付される SAF の一部であるので、RACF 以外のお客様にもご利用いただけます。

RSE_JAVAOPTS

追加の RSE 固有の Java オプション。この定義の詳細については、46 ページの『_RSE_JAVAOPTS での追加 Java 始動パラメーターの定義』を参照してください。

デフォルトでは、Developer for System z は ISPF の TSO/ISPF クライアント・ゲートウェイを TSO コマンド・サービスに使用します。次の _RSE_JAVAOPTS オプションをコメント解除した場合は、APPC トランザクションが代わりに使用されます。

```
RSE_JAVAOPTS="$_RSE_JAVAOPTS -DTSO_SERVER=APPC"
```

TSO コマンド・サービス、SCLM Developer Toolkit、または CARMA に ISPF の TSO/ISPF クライアント・ゲートウェイが使用される場合は、以下の定義が必要です。

_CMDSERV_BASE_HOME

TSO/ISPF クライアント・ゲートウェイ・サービスを提供する ISPF コードのホーム・ディレクトリー。デフォルトは /usr/lpp/ispf です。使用する ISPF インストール済み環境に合わせて変更してください。このディレクトィブは、ISPF の TSO/ISPF クライアント・ゲートウェイを使用する場合にのみ必要です。

_CMDSERV_CONF_HOME

ISPF 基本構成ディレクトリー。デフォルトは /etc/rdz です。TSO/ISPF クライアント・ゲートウェイ・カスタマイズ・ファイル、ISPF.conf のロケーションに合わせて変更してください。このディレクティブは、ISPF の TSO/ISPF クライアント・ゲートウェイを使用する場合にのみ必要です。

_CMDSERV_WORK_HOME

ISPF 基本作業ディレクトリー。デフォルトは /var/rdz です。TSO/ISPF クライアント・ゲートウェイが使用する WORKAREA ディレクトリーのロケーションに合わせて変更してください。このディレクティブは、ISPF の TSO/ISPF クライアント・ゲートウェイを使用する場合にのみ必要です。

注:

- _CMDSERV_WORK_HOME で指定したパスには、TSO/ISPF クライアント・ゲートウェイによって /WORKAREA が追加されます。手動では追加しないでください。
- カスタマイズ可能な環境の構築に SFEKSAMP(FEKSETUP) サンプル・ジョブを使用しなかった場合は、_CMDSERV_WORK_HOME で指定したパスに WORKAREA ディレクトリーが存在することを確認してください。ディレクトリー許可ビットは 777 でなければなりません。

STEPLIB

STEPLIB については、以前に、必要な定義のセクションで説明しました。

RSE_CMDSERV_OPTS

追加の TSO/ISPF クライアント・ゲートウェイ固有の Java オプション。デフォルトは "" です。この定義の詳細については、51 ページの

『RSE_CMDSERV_OPTS での追加 Java 始動パラメーターの定義』を参照してください。このディレクティブは、ISPF の TSO/ISPF クライアント・ゲートウェイを使用する場合にのみ必要です。

SCLM Developer Toolkit を使用する場合は、以下の定義が必要です。

SCLMDT_CONF_HOME

SCLM Developer Toolkit 基本構成ディレクトリー。デフォルトは /var/rdz/sclmdt です。SCLMDT が SCLM プロジェクト情報を保管するために使用する CONFIG ディレクトリーのロケーションに合わせて変更してください。このディレクティブは、SCLMDT を使用する場合にのみ必要です。

注: SCLMDT_CONF_HOME に指定したパスには、SCLMDT によって /CONFIG と /CONFIG/PROJECT が追加されます。手動では追加しないでください。

STEPLIB

STEPLIB については、以前に、必要な定義のセクションで説明しました。

SCLMDT_TRANTABLE

ロング/ショート・ネーム変換 VSAM の名前。デフォルトは FEK.#CUST.LSTRANS.FILE です。コメント解除し、SCLM サンプル・ジョブ ISP.SISPSAMP(FLM02LST) で使用される名前に合わせて変更してください。このディレクティブが必要になるのは、SCLM Developer Toolkit でロング/ショート・ネーム変換を使用する場合だけです。

ANT_HOME

Ant インストールのホーム・ディレクトリー。デフォルトは /usr/lpp/apache/Ant/apache-ant-1.7.1 です。使用する Ant インストールに合わせて変更してください。このディレクティブが必要になるのは、SCLM Developer Toolkit で JAVA/J2EE ビルド・サポートを使用する場合だけです。

以下の定義はオプションです。省略した場合は、デフォルト値が使用されます。

_RSE_PORTRANGE

RSE サーバーがクライアントとの通信用に開くことができるポート範囲を指定します。デフォルトでは、任意のポートを使用できます。この定義の詳細については、45 ページの『RSE サーバーに使用可能な PORTRANGE の定義』を参照してください。これは、オプションのディレクティブです。

_BPXK_SETIBMOPT_TRANSPORT

使用する TCP/IP スタックの名前を指定します。デフォルトは TCPIP です。コメント解除して、要求された TCPIP スタック名に変更してください。この名前は、関連する TCPIP.DATA 内の TCPIPJOBNAME ステートメントで定義されています。これは、オプションのディレクティブです。

注:

- サーバー JCL 内の SYSTCPD DD ステートメントのコーディングでは、要求されたスタックのアフィニティーは設定されません。
- このディレクティブがアクティブでないときは、RSE がシステム上の使用可能なすべてのスタックにバインドします (BIND INADDRANY)。

_FEKFSCMD_TP_NAME_

APPC トランザクション・プログラム名。デフォルト値は FEKFRSRV です。APPC トランザクションを定義するときにデフォルトのトランザクション・プログラム名を使用しなかった場合は、この定義をコメント解除して変更してください。これは、オプションのディレクティブです。

_FEKFSCMD_PARTNER_LU_

RSE サーバーに、この APPC パートナー LU の使用を強制します。デフォルトは、APPC の構成時に指定した 基本 LU です。これは、オプションのディレクティブです。

GSK_CRL_SECURITY_LEVEL

証明書の妥当性検査で、取り消された証明書がないかどうか CRL を検査するために、SSL アプリケーションが LDAP サーバーとの接続時に使用するセキュリティのレベルを指定します。デフォルトは MEDIUM です。指定した値を強制的に使用させるには、コメント解除して変更します。これは、オプションのディレクティブです。以下の値が有効です。

- LOW - LDAP サーバーに接続できない場合でも、証明書の妥当性検査は失敗しません。
- MEDIUM - 証明書の妥当性検査を行うには LDAP サーバーが接続可能であることが必要ですが、CRL を定義する必要はありません。これはデフォルトです。

- HIGH - 証明書の妥当性検査を行うには LDAP サーバーが接続可能であり、CRL を定義する必要があります。

注: このディレクティブには z/OS 1.9 以上が必要です。

GSK_LDAP_SERVER

1 つ以上の LDAP サーバー・ホスト名をブランクで区切って指定します。指定した LDAP サーバーを強制的に使用させて、それらのサーバーの CRL を取得するには、コメント解除して変更します。これは、オプションのディレクティブです。

ホスト名は TCP/IP アドレスと URL のどちらでもかまいません。それぞれのホスト名は、コロン (:) でホスト名と区切った、オプションのポート番号を含むことができます。

GSK_LDAP_PORT

LDAP サーバー・ポートを指定します。デフォルトは 389 です。指定した値を強制的に使用させるには、コメント解除して変更します。これは、オプションのディレクティブです。

GSK_LDAP_USER

LDAP サーバーに接続するときに使用する識別名を指定します。指定した値を強制的に使用させるには、コメント解除して変更します。これは、オプションのディレクティブです。

GSK_LDAP_PASSWORD

LDAP サーバーに接続するときに使用するパスワードを指定します。指定した値を強制的に使用させるには、コメント解除して変更します。これは、オプションのディレクティブです。

以下の定義は必須であり、IBM サポートから指示された場合以外、変更しないでください。

_CEE_RUNOPTS

言語環境プログラム (LE) ランタイム・オプション。デフォルトは "ALL31(ON) HEAP(32M,32K,ANYWHERE,KEEP,,) TRAP(ON)" です。変更しないでください。

_BPX_SHAREAS

シェルと同じアドレス・スペースでフォアグラウンド・プロセスを実行します。デフォルトは YES です。変更しないでください。

_BPX_SPAWN_SCRIPT

シェル・スクリプトを spawn() 関数から直接実行します。デフォルトは YES です。変更しないでください。

JAVA_PROPAGATE

スレッド作成時にセキュリティーおよびワークロード・コンテキストを伝搬します (Java バージョン 1.4 以前専用)。デフォルトは NO です。変更しないでください。

RSE_LIB

RSE ライブラリー・パス。デフォルトは \$RSE_HOME/lib です。変更しないでください。

PATH コマンド・パス。デフォルトは `.:$JAVA_HOME/bin:$RSE_HOME/bin:$_CMDSERV_BASE_HOME/bin:$PATH` です。変更しないでください。

LIBPATH

ライブラリー・パス。デフォルトは長すぎるので、ここには繰り返しません。変更しないでください。

CLASSPATH

クラス・パス。デフォルトは長すぎるので、ここには繰り返しません。変更しないでください。

_RSE_CMDSERV_OPTS

追加の TSO コマンド・サービス固有の Java オプション。デフォルトは `"&SESSION=SPAWN$_RSE_CMDSERV_OPTS"` です。変更しないでください。

_RSE_JAVAOPTS

追加の RSE 固有の Java オプション。デフォルトは長すぎるので、ここには繰り返しません。変更しないでください。

_RSE_SERVER_CLASS

RSE サーバーの Java クラス。デフォルトは `org.eclipse.dstore.core.server.Server` です。変更しないでください。

_RSE_DAEMON_CLASS

RSE デーモンの Java クラス。デフォルトは `com.ibm.etools.zos.server.RseDaemon` です。変更しないでください。

_RSE_POOL_SERVER_CLASS

RSE スレッド・プールの Java クラス。デフォルトは `com.ibm.etools.zos.server.ThreadPoolProcess` です。変更しないでください。

_RSE_LOCKD_CLASS

RSE ロック・デーモンの Java クラス。デフォルトは `com.ibm.ftt.rse.mvs.server.miners.MVSLockDaemon` です。変更しないでください。

_RSE_SERVER_TIMEOUT

RSE サーバーの (クライアントを待つ) タイムアウト値 (ミリ秒単位)。デフォルトは 120000 (2 分) です。変更しないでください。

SCLMDT_BASE_HOME

SCLM Developer Toolkit コードのホーム・ディレクトリー。デフォルトは `$RSE_HOME` です。変更しないでください。

SCLMDT_WORK_HOME

SCLM Developer Toolkit 基本作業ディレクトリー。デフォルトは `$_CMDSERV_WORK_HOME` です。変更しないでください。

CGI_DTWORK

古いクライアント用の SCLM Developer Toolkit サポート。デフォルトは `$_SCLMDT_WORK_HOME` です。変更しないでください。

RSE サーバーに使用可能な PORTRANGE の定義

これは、RSE サーバーがクライアントと通信できるポートを指定する、`rsed.envvars` カスタマイズの一環です。このポート範囲は、RSE デーモン・ポートとは関係ありません。

ポートの使用法を理解しやすいように、以下で RSE の接続プロセスを簡単に説明します。

1. クライアントは、ホスト・ポート 4035、RSE デーモンに接続します。
2. RSE デーモンは、RSE サーバー・スレッドを作成します。
3. RSE サーバーは、クライアントが接続するホスト・ポートを開きます。このポートの選択はユーザーが構成でき、クライアント上でサブシステム・プロパティ・タブによって構成するか (これはお勧めできません)、`rsed.envvars` 内の `_RSE_PORTRANGE` 定義を通じて構成します。
4. RSE デーモンは、クライアントにポート番号を返します。
5. クライアントは、ホスト・ポートに接続します。

注:

- このプロセスは、REXEC/SSH を使用した (オプションの) 代替接続方式とほとんど同じです。
- 詳細については、205 ページの『第 11 章 Developer for System z について』を参照してください。

クライアントが z/OS と通信できるように、ポート範囲を指定するには、`rsed.envvars` 内で次の行をコメント解除し、カスタマイズします。

```
#_RSE_PORTRANGE=8108-8118
```

注: ポート範囲を選択する前に、**NETSTAT** および **NETSTAT PORTL** コマンドで、その範囲がシステム上で使用可能であることを確認してください。

PORTRANGE のフォーマットは、次のとおりです。`_RSE_PORTRANGE=min-max` (max は、その値を含みません。例えば、`_RSE_PORTRANGE=8108-8118` は 8108 から 8117 までのポート番号が使用できることを意味します)。RSE サーバーによって使用されるポート番号は、次の順序で決定されます。

1. ゼロ以外のポート番号がクライアント上のサブシステム・プロパティで指定されている場合は、指定したポート番号が使用されます。そのポートが使用不可の場合、接続は失敗します。このセットアップは、お勧めできません。

注: ホストでは、`rsed.envvars` に `deny.nonzero.port=true` ディレクティブを指定することで、このタイプの接続要求を拒否できます。このディレクティブの詳細については、46 ページの『`_RSE_JAVAOPTS` での追加 Java 始動パラメーターの定義』を参照してください。

2. サブシステム・プロパティ内のポート番号が 0 の場合、しかも `_RSE_PORTRANGE` が `rsed.envvars` で指定されている場合は、`_RSE_PORTRANGE` で指定されたポート範囲が使用されます。範囲内で使用可能なポートがない場合、接続は失敗します。

3. サブシステム・プロパティ内のポート番号が 0 で、`_RSE_PORTRANGE` が `rsed.envvars` の中で指定されていない場合は、使用可能な任意のポートが使用されます。

注: サーバーがポートを開き、listen している場合、そのポート番号を別のサーバーが使用することはできません。しかし、接続された後は、同じポート番号を再度使用できます。これは、同時に接続するユーザーの数が、範囲内のポートの数によって制限されないことを意味します。

`_RSE_JAVAOPTS` での追加 Java 始動パラメーターの定義

さまざまな `_RSE_*OPTS` ディレクティブにより、`rsed.envvars` は RSE プロセスの始動時に、Java に追加パラメーターを指定できます。`rsed.envvars` に含まれているサンプル・オプションは、それらをコメント解除することによってアクティブにできます。

`_RSE_JAVAOPTS` は、標準および RSE 固有の Java オプションを定義します。

`_RSE_JAVAOPTS=""`

変数初期化。変更しないでください。

`_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Xms1m -Xmx256m"`

初期 (Xms) および最大 (Xmx) ヒープ・サイズを設定します。デフォルトはそれぞれ、1M と 256M です。希望するヒープ・サイズ値を強制的に使用させるには、変更してください。このディレクティブがコメント化されている場合は、Java のデフォルト値が使用されます。デフォルト値はそれぞれ 4M と 512M です (Java 5.0 の場合は 1M と 64M)。

注: このディレクティブの最適値を判別するには、247 ページの『主要なリソース定義』を参照してください。

`_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Ddaemon.log=/var/rdz/logs"`

RSE デーモンおよびサーバーのログと RSE 監査データを保持するディレクトリ。デフォルトは `/var/rdz/logs` です。希望するロケーションを強制的に使用させるには、変更してください。このディレクティブがコメント化されている場合は、RSE デーモンに割り当てたユーザー ID のホーム・ディレクトリが使用されます。ホーム・ディレクトリはユーザー ID の OMVS セキュリティー・セグメントで定義されます。

注: このディレクティブ (または対応するホーム・ディレクトリ) で絶対パスが指定されていない (パスがスラッシュ (/) で始まっていない) 場合、実際のログ・ロケーションは、構成ディレクトリ (デフォルトでは `/etc/rdz`) に対して相対的になります。

`_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Duser.log=/var/rdz/logs"`

ユーザー固有のログがあるディレクトリ。デフォルトは `/var/rdz/logs` です。希望するロケーションを強制的に使用させるには、変更してください。このディレクティブがコメント化されている場合は、クライアント・ユーザー ID のホーム・ディレクトリが使用されます。ホーム・ディレクトリはユーザー ID の OMVS セキュリティー・セグメントで定義されます。

注:

- このディレクティブ (または対応するホーム・ディレクトリー) で絶対パスが指定されていない (パスがスラッシュ (/) で始まっていない) 場合、実際のログ・ロケーションは、構成ディレクトリー (デフォルトでは /etc/rdz) に対して相対的になります。
- ユーザー・ログの完全なパスは、userlog/dstorelog/\$LOGNAME/ です。ここで、userlog は user.log ディレクティブの値、dstorelog は DSTORE_LOG_DIRECTORY ディレクティブの値、\$LOGNAME は大文字で表記されたクライアントのユーザー ID です。
- 各クライアントが \$LOGNAME を作成できるように、userlog/dstorelog の許可ビットが設定されていることを確認してください。

_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -DDSTORE_LOG_DIRECTORY="

このディレクトリーは、user.log ディレクティブに指定されているパスに付加されます。ディレクトリーが付加されたパスは、ユーザー固有のログへのパスとなります。デフォルトはヌル・ストリングです。指定したディレクトリーを強制的に使用させるには、変更してください。このディレクティブがコメント化されている場合は、.eclipse/RSE/ が使用されます。

注:

- ユーザー・ログの完全なパスは、userlog/dstorelog/\$LOGNAME/ です。ここで、userlog は user.log ディレクティブの値、dstorelog は DSTORE_LOG_DIRECTORY ディレクティブの値、\$LOGNAME は大文字で表記されたクライアントのユーザー ID です。
- ここで指定されたディレクトリーは、user.log に指定されたディレクトリーに相対的であるため、スラッシュ (/) で始まっていない場合があります。
- 各クライアントが \$LOGNAME を作成できるように、userlog/dstorelog の許可ビットが設定されていることを確認してください。

以下のディレクティブは、デフォルトではコメント化されています。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -Dmaximum.clients=60"

1 つのスレッド・プールでサービスできるクライアントの最大数。デフォルトは 60 です。1 スレッド・プール当たりのクライアント数を制限するには、コメント解除してカスタマイズします。他の制限のために、RSE がこの限度に到達しない場合もあることに注意してください。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -Dmaximum.threads=1000"

新規クライアントを許可するための、1 つのスレッド・プールでのアクティブ・スレッド数の最大値。デフォルトは 1000 です。1 スレッド・プール当たりのクライアント数を、使用中のスレッド数に基づいて制限するには、コメント解除してカスタマイズします。それぞれのクライアント接続が複数 (16 以上) のスレッドを使用すること、および他の制限のために RSE がこの限度に到達しない場合があることに注意してください。

注: この値は、SYS1.PARMLIB(BPXPRMxx) での MAXTHREADS および MAXTHREADTASKS の設定より小さくする必要があります。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -Dminimum.threadpool.process=1"

アクティブ・スレッド・プールの最小数。デフォルトは 1 です。少なくともリストされた数のスレッド・プール・プロセスを開始するには、コメント解除してカスタマイズします。スレッド・プール・プロセスは、RSE サーバー・スレッドのロード・バランシングに使用されます。より多くの新規プロセスが必要になった場合は、その時点で新規プロセスが開始されます。前もって新規プロセスを開始しておく、接続遅延を回避できますが、アイドル時間に使用されるリソースが増えます。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -Dmaximum.threadpool.process=100"

アクティブ・スレッド・プールの最大数。デフォルトは 100 です。スレッド・プール・プロセスの数を制限するには、コメント解除してカスタマイズします。スレッド・プール・プロセスは、RSE サーバー・スレッドのロード・バランシングに使用されます。このため、スレッドを制限すると、アクティブなクライアント接続の数も制限されます。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -Dipv6=true"

TCP/IP バージョン。デフォルトは false で、IPv4 インターフェースが使用されることを意味します。IPv6 インターフェースを使用するには、コメント解除して true を指定します。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -Dkeep.last.log=true"

前のセッションに属するホスト・ログ・ファイルのコピーを保持します。デフォルトは false です。サーバーを始動してクライアントに接続するときに、前のログ・ファイルの名前を *.last に変更する場合は、コメント解除して true を指定します。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -Denable.standard.log=true"

スレッド・プールの STDOUT および STDERR ストリームをログ・ファイルに書き込みます。デフォルトは false です。STDOUT および STDERR ストリームを保存する場合は、コメント解除して true を指定します。結果のログ・ファイルは、daemon.log ディレクティブで参照されるディレクトリーに置かれます。

注:

- **MODIFY RSESTANDARDLOG** オペレーター・コマンドを使用して、ストリーム・ログ・ファイルの更新を動的に開始または停止することができます。
- **enable.standard.log** ディレクティブがアクティブであるときは、ユーザー固有の stdout.log および stderr.log ログ・ファイルが存在しません。ユーザー固有のデータは、対応する RSE スレッド・プール・ストリームに書き込まれます。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -Denable.port.of.entry=true"

Port Of Entry (POE) 検査オプション。デフォルトは false です。クライアント接続の POE 検査を強制的に行わせるには、コメント解除して true を指定します。POE 検査のとき、クライアントの IP アドレスは、ご使用のセキュリティー・ソフトウェアによってネットワーク・アクセス・セキュリティー・ゾーンにマップされます。クライアント・ユーザー ID は、セキュリティー・ゾーンを定義しているプロファイルを使用する権限を持っている必要があります。

注:

- POE 検査は、使用しているセキュリティー製品でも有効にする必要があります。
- POE 検査を有効にすると、INETD など、他の z/OS UNIX にも有効になります。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -Denable.certificate.mapping=false"

ご使用のセキュリティー・ソフトウェアを使用して、X.509 証明書でログオンを認証します。デフォルトは true です。セキュリティー・ソフトウェアの X.509 サポートに依存せず、RSE デーモンに認証を行わせるには、コメント解除して false を指定します。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -Denable.automount=true"

z/OS UNIX 自動マウントで作成されたホーム・ディレクトリーをサポートします。デフォルトは false です。z/OS UNIX 自動マウントでクライアント・ユーザー ID がディレクトリーの所有者として使用されるようにするには、コメント解除して true を指定します。

注: z/OS UNIX 自動マウントでは、ファイル・システムを作成する際にサービスを起動したプロセスのユーザー ID が使用されます。このオプションが使用不可になっている場合、このプロセスは RSE スレッド・プール・サーバー (ユーザー ID STCRSE) です。このオプションを使用可能になっている場合は、サービスを起動する前に、クライアント・ユーザー ID を使用して新しい一時プロセスが作成されます。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -Denable.audit.log=true"

監査オプション。デフォルトは false です。クライアントが実行したアクションの監査ロギングを強制的に行わせるには、コメント解除して true を指定します。監査ログは、RSE デーモンのログ・ロケーションに書き込まれます。その場所を知るには、_RSE_JAVAOPTS 変数の daemon.log オプションを参照してください。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -Daudit.cycle=30"

1 つの監査ログ・ファイルに保管される日数。デフォルトは 30 です。1 つの監査ログ・ファイルに書き込む監査データの量を制御するには、コメント解除してカスタマイズします。最大値は 365 です。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -Daudit.retention.period=0"

監査ログが保持される日数。デフォルトは 0 (制限なし) です。一定の日数を経た後に監査ログを削除するには、コメント解除してカスタマイズします。最大値は 365 です。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -Ddeny.nonzero.port=true"

クライアントによる通信ポート番号の選択を禁止します。デフォルトは false です。RSE サーバーが接続にどのホスト・ポートを使用しなければならないかをクライアントで指定される通信を拒否するには、コメント解除して true を指定します。詳細については、45 ページの『RSE サーバーに使用可能な PORTRANGE の定義』を参照してください。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -Dsingle.logon=false"

1 つのユーザー ID による複数回のログオンを禁止します。デフォルトは

true です。1 つのユーザー ID が 1 つの RSE デーモンに複数回ログオンできるようにするには、コメント解除して false を指定します。

注: このディレクティブがアクティブでないか、または false に設定されていない場合は、2 回目にログオンが試行されると 1 回目のログオン試行がホストによって取り消されます。この取り消しの際に、コンソール・メッセージ FEK210I が表示されます。

RSE_JAVAOPTS="\$_RSE_JAVAOPTS -Dprocess.cleanup.interval=0"

リカバリー不能エラーの状態にある RSE スレッド・プールを自動的に除去します。デフォルトでは、エラー状態のスレッド・プールは自動的に除去されません。エラー状態の RSE スレッド・プールを一定の時間間隔 (秒単位) で自動的に除去するには、コメント解除してカスタマイズします。0 を指定すると、この機能は使用不可になります。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -DAPPLID=FEKAPPL"

RSE サーバー・アプリケーション ID。デフォルトは FEKAPPL です。希望するアプリケーション ID を強制的に使用させるには、このオプションをコメント解除してカスタマイズします。

注:

- 使用するセキュリティー・ソフトウェアに対してアプリケーション ID を定義する必要があります。そうしないと、クライアントはログオンできません。
- この値を変更した場合のセキュリティーへの影響については、175 ページの『PassTicket の使用』を参照してください。
- このアプリケーション ID は、JES ジョブ・モニターが使用するアプリケーション ID と一致している必要があります。JES ジョブ・モニターに対してアプリケーション ID を定義する方法については、30 ページの『FEJCNFG、JES ジョブ・モニター構成ファイル』を参照してください。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -DDENY_PASSWORD_SAVE=true"

パスワード保存オプション。デフォルトは false です。ユーザーがホスト・パスワードをクライアントに保存できないようにするには、コメント解除して true を指定します。以前に保存されていたパスワードは、除去されます。このオプションは、クライアントのバージョン 7.1 以上でのみ機能します。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -Dhide_zos_unix=true"

z/OS UNIX オプションを隠します。デフォルトは false です。クライアントで z/OS UNIX のエレメント (ディレクトリー構造とコマンド行) が表示されないようにするには、コメント解除して true を指定します。このオプションは、クライアントのバージョン 7.6 以上でのみ機能します。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS

-DDSTORE_IDLE_SHUTDOWN_TIMEOUT=3600000"

活動停止中のクライアントを切断します。デフォルトでは、活動停止中のクライアントは切断されません。リストされたミリ秒数 (3600000 が 1 時間に相当) の間、活動を停止していたクライアントを切断するには、コメント解除してカスタマイズします。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -DDSTORE_TRACING_ON=true"

dstore トレースを開始します。IBM サポートから指示された場合にのみ使用してください。結果の .dstoreTrace ログ・ファイルは、EBCDIC ではなく Unicode (ASCII) で作成されます。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -DDSTORE_MEMLOGGING_ON=true"

dstore メモリー・トレースを開始します。IBM サポートから指示された場合にのみ使用してください。結果の .dstoreMemLogging ログ・ファイルは、EBCDIC ではなく Unicode (ASCII) で作成されます。

#_RSE_JAVAOPTS="\$_RSE_JAVAOPTS -DTSO_SERVER=APPC"

TSO コマンド・サービスに APPC トランザクションを使用します。デフォルトでは、ISPF の TSO/ISPF クライアント・ゲートウェイが使用されます。代わりに APPC トランザクションを使用するには、コメント解除します。割り当てられている値を変更しないでください。

_RSE_CMDSERV_OPTS での追加 Java 始動パラメーターの定義

さまざまな **_RSE_*OPTS** ディレクティブにより、**rsed.envvars** は RSE プロセスの始動時に、Java に追加パラメーターを指定できます。**rsed.envvars** に含まれているサンプル・オプションは、それらをコメント解除することによってアクティブにできます。

_RSE_CMDSERV_OPTS ディレクティブは RSE 固有の Java オプションであり、ISPF の TSO/ISPF クライアント・ゲートウェイが Developer for System z によって使用されている場合にのみ有効です (これはデフォルトです)。

_RSE_CMDSERV_OPTS=""

変数初期化。変更しないでください。

_RSE_CMDSERV_OPTS="\$_RSE_CMDSERV_OPTS &ISPROF=&SYSUID..ISPROF="

ISPF の初期化に既存の ISPF プロファイルを使用します。指定した ISPF プロファイルを使用するには、データ・セット名をコメント解除して変更します。

データ・セット名の中で、以下の変数を使用できます。

- **&SYSUID** (開発者のユーザー ID の代わりに使用)
- **&SYSPREF** (開発者の TSO 接頭部の代わりに使用)

ISPF.conf、ISPF の TSO/ISPF クライアント・ゲートウェイ構成ファイル

ISPF の TSO/ISPF クライアント・ゲートウェイは、ISPF.conf 内の定義を使用して、バッチ TSO および ISPF コマンドを実行するための有効な環境を作成します。Developer for System z は、その環境を使用して、一部の MVS ベースのサービスを実行します。それらのサービスには、TSO コマンド・サービス、SCLM Developer Toolkit サービス、および代替 CARMA 始動方式が含まれます。

ISPF.conf は /etc/rdz/ に置かれます。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。このファイルは、TSO **OEDIT** コマンドで編集できます。

US コード・ページを使用する場合、コメント行はアスタリスク (*) で始まります。データ行には、ディレクティブとそれに割り当てられる値のみを入れることができます。同じ行にコメントを入れることはできません。行の継続はサポートされていません。データ・セット名を連結するときは、それらを同じ行に追加し、名前同士をコンマ (,) で分離します。

ISPF データ・セットの正しい名前を指定するだけでなく、次の例に示すように、TSO コマンド・サービス・データ・セット名 FEK.SFEKPROC を SYSPROC ステートメントまたは SYSEXEC ステートメントに追加する必要があります。

```
* REQUIRED:
sysproc=ISP.SISPLIB,FEK.SFEKPROC
isplib=ISP.SISPMENU
isptlib=ISP.SISPTENU
ispplib=ISP.SISPPENU
ispslib=ISP.SISPSLIB
ispllib=ISP.SISLOAD

* OPTIONAL:
*allocjob = FEK.#CUST.CNTL(CRAISPRX)
*ISPF_timeout = 900
```

図9. ISPF.conf - ISPF 構成ファイル

注:

- TSO 環境をカスタマイズするために、独自の DD のようなステートメントとデータ・セット連結を追加し、TSO ログオン・プロシーチャーを模倣することができます。詳細については、283 ページの『第 16 章 TSO 環境のカスタマイズ』を参照してください。
- TSO/ISPF クライアント・ゲートウェイは、**ISPSTART** などの ISPF コマンドをインターセプトする (サード・パーティーの) 製品を使用した場合、正しく機能しない可能性があります。その製品を Developer for System z に対して無効にする方法については、製品の資料を参照してください。その製品が特定の DD ステートメントを DUMMY に割り振ることを必要とする場合は、ISPF.conf の中で、その DD ステートメントを nullfile に割り振ることによってシミュレートできます。

次に例を示します。

```
ISPTRACE=nullfile
```

- allocjob ディレクティブを使用する場合は、前に ISPF.conf で行った DD 定義を元に戻さないように注意してください。
- SMFPRMxx parmlib メンバー内の JWT パラメーターの設定が ISPF.conf 内の ISPF_timeout 値よりも小さい場合は、モジュール ISPZTSO のシステム異常終了 522 が予想されます。これによって、Developer for System z の操作が影響を受けることはありません。必要なときには TSO/ISPF クライアント・ゲートウェイが自動的に再始動されるからです。
- 変更は、すべての新規呼び出しについてアクティブになります。サーバーの再始動は必要ありません。

オプションのコンポーネント

上記のカスタマイズ・ステップは、基本的な Developer for System z セットアップ用です。オプション・コンポーネントのカスタマイズ要件については、それらのコンポーネントに関する章を参照してください。

- 55 ページの『第 3 章 (オプション) 共通アクセス・リポジトリ・マネージャー (CARMA)』
- 79 ページの『第 4 章 (オプション) Application Deployment Manager』
- 87 ページの『第 5 章 (オプション) SCLM Developer Toolkit』
- 95 ページの『(オプション) DB2 ストアード・プロシージャ』
- 98 ページの『(オプション) CICS 双方向言語サポート』
- 100 ページの『(オプション) RSE SSL 暗号化』
- 102 ページの『(オプション) RSE トレース』
- 104 ページの『(オプション) ホスト・ベース・プロパティ・グループ』
- 105 ページの『(オプション) ホスト・ベース・プロジェクト』
- 106 ページの『(オプション) File Manager Integration』
- 108 ページの『(オプション) 編集不可能文字』
- 109 ページの『(オプション) REXEC (または SSH) の使用』
- 111 ページの『(オプション) TSO コマンド・サービス用の APPC トランザクション』
- 115 ページの『(オプション) WORKAREA クリーンアップ』

インストール検査

各種インストール検査プログラム (IVP) については、117 ページの『第 7 章 インストール検査』で説明しています。これは、一部の IVP がオプション・コンポーネント用であるためです。

第 3 章 (オプション) 共通アクセス・リポジトリ・マネージャー (CARMA)

共通アクセス・リポジトリ・マネージャー (CARMA) は、Repository Access Manager (RAM) を作成する開発者向けの生産性援助機能です。RAM は、z/OS ベースの Software Configuration Manager (SCM) 用のアプリケーション・プログラミング・インターフェース (API) です。

ユーザー作成アプリケーションは CARMA サーバーを始動でき、CARMA サーバーは RAM をロードし、SCM にアクセスする標準インターフェースを提供します。

Developer for System z は CARMA サーバーを始動する複数の方式をサポートしており、それぞれの方式には独自の利点と欠点があります。

- 「バッチ実行依頼」方式は、ジョブを実行依頼することによって CARMA サーバーを始動します。これが、提供されたサンプル構成ファイルで 사용되는デフォルトの方式です。この方式の利点は、ジョブ出力内で CARMA ログに簡単にアクセスできることです。また、開発者自身が保守する開発者ごとのカスタム・サーバー JCL を使用できます。ただし、この方式では、CARMA サーバーを始動した開発者ごとに 1 つずつ JES イニシエーターが使用されます。
- 「CRASTART」方式は、CARMA サーバーを RSE 内のサブタスクとして始動します。この方式では、CARMA サーバーを始動するために必要なデータ・セット割り振りとプログラム呼び出しを別個の構成ファイルで定義し、その構成ファイルを使用するので、非常に柔軟なセットアップが可能です。この方式では最良のパフォーマンスが得られ、使用するリソースも最少で済みますが、モジュール CRASTART を LPA 内に配置する必要があります。
- 「TSO/ISPF クライアント・ゲートウェイ」方式は、ISPF の TSO/ISPF クライアント・ゲートウェイを使用して TSO 環境または ISPF 環境を作成し、その中で CARMA サーバーを始動します。この方式では、ISPF.conf の機能を使用して柔軟なデータ・セット割り振りができます。ただし、この方式は通常の TSO または ISPF 操作に干渉する SCM へのアクセスには適していません。

要件およびチェックリスト

このカスタマイズ・タスクを完了するには、セキュリティー管理者および TCP/IP 管理者の支援が必要になります。このタスクには、以下のリソースまたは特殊なカスタマイズ・タスクが必要です。

- 内部通信用の TCP/IP ポート範囲
- 開発者に CARMA VSAM ファイルの更新を許可するセキュリティー規則
- (オプション) ユーザーに CRA* ジョブの実行依頼を許可するセキュリティー規則
- (オプション) LPA 更新

ご使用のサイトで CARMA の使用を開始するには、以下のタスクを行う必要があります。特に断りがない限り、すべてのタスクは必須です。

1. 必要な CARMA コンポーネントを作成します。詳細については、『CARMA コンポーネント』を参照してください。
2. CARMA とインターフェースするための RSE 構成ファイルの初期カスタマイズ。完全なカスタマイズは、CARMA を始動するために選択した方式によって異なります。詳細については、58 ページの『CARMA への RSE インターフェース』を参照してください。
3. CARMA を始動する方式を選択し、関連する構成ファイルに必要なカスタマイズを行います。詳細については、以下を参照してください。
 - 60 ページの『バッチ実行依頼を使用した CARMA サーバーの始動』
 - 62 ページの『(オプション) CRASTART を使用した代替の CARMA サーバーの始動』
 - 65 ページの『(オプション) TSO/ISPF クライアント・ゲートウェイを使用した代替の CARMA サーバーの始動』
4. オプションとして、サンプルの Repository Access Manager (RAM) をアクティブにします。詳細については、67 ページの『(オプション) サンプルの Repository Access Manager (RAM) のアクティブ化』を参照してください。
5. オプションとして、CA Endeavor® RAM をアクティブにします。詳細については、68 ページの『(オプション) CA Endeavor® SCM RAM のアクティブ化』を参照してください。
6. オプションとして、IRXJCL に代わる CRAXJCL を作成します。詳細については、77 ページの『(オプション) IRXJCL と CRAXJCL』を参照してください。

注: この章で参照するサンプルのメンバーは FEK.#CUST.* および /etc/rdz の中に置かれます。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

CARMA コンポーネント

以下の CARMA コンポーネントは、選択する始動方式に関係なく、必ずカスタマイズしなければなりません。参照するサンプルのメンバーは、FEK.#CUST.JCL に置かれます。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

1. FEK.#CUST.JCL(CRA\$VDEF) JCL をカスタマイズして実行依頼します。カスタマイズの手順については、CRA\$VDEF 内のドキュメンテーションを参照してください。CRA\$VDEF は、CARMA 構成 VSAM データ・セット CRADEF の作成と事前準備を行います。
2. FEK.#CUST.JCL(CRA\$VMSG) JCL をカスタマイズして実行依頼します。カスタマイズの手順については、CRA\$VMSG 内のドキュメンテーションを参照してください。CRA\$VMSG は、CARMA メッセージ VSAM データ・セット CRAMSG の作成と事前準備を行います。
3. FEK.#CUST.JCL(CRA\$VSTR) JCL をカスタマイズして実行依頼します。カスタマイズの手順については、CRA\$VSTR 内のドキュメンテーションを参照してください。CRA\$VSTR は、CARMA カスタム情報 VSAM データ・セット CRASTRS の作成と事前準備を行います。

注:

- これらのジョブで作成される CARMA VSAM は、サンプルの RAM を定義します。CA Endeavor® RAM を定義する場合は、68 ページの『(オプション) CA Endeavor® SCM RAM のアクティブ化』を参照してください。
- (カスタムの) RAM の定義を既存の VSAM 構成にマージする必要がある場合は、サンプル・ジョブ FEK.#CUST.JCL(CRA#UADD) を参照してください。このジョブは、変更する CARMA VSAM ごとにカスタマイズし、実行依頼する必要があります。各種の CARMA VSAM が使用するレコード構造の詳細については、「*Rational Developer for System z Common Access Repository Manager Developer's Guide*」(SC23-7660) を参照してください。
- VSAM から順次データ・セットへアクティブな定義を抽出するには、サンプル・ジョブ FEK.#CUST.JCL(CRA#UQRY) を使用します。

CARMA VSAM のマイグレーションに関する注

Developer for System z バージョン 7.6.1 は、メッセージ長の制限を解消するために、CARMA カスタム情報 VSAM データ・セット CRASTRS の新しいデータ構造レイアウトをサポートしています。

Developer for System z バージョン 7.6.1 までは、CARMA カスタム情報 VSAM データ・セットで定義されるストリングが、事前に定義された長さに制限されます。この制限のために、RAM 開発者は記述ストリングを短くするか、フルサイズのストリングを表示するためにクライアント・サイド・プラグインを使用する必要があります。

Developer for System z バージョン 7.6.1 は、CARMA カスタム情報 VSAM データ・セット CRASTRS の新しい可変長データ構造レイアウトをサポートしています。このレイアウトでは、ストリングが固定長とならずに区切り文字で分離されます。

既存の固定長の CARMA カスタム情報 VSAM データ・セット CRASTRS を新しい可変長フォーマットに変換するには、FEK.SFEKSAMP(CRA#VS2) JCL をカスタマイズして実行依頼します。

注:

- バージョン 7.6.1 より、サンプルの CARMA カスタム情報 VSAM データ・セットは可変長フォーマットで出荷されます。
- バージョン 7.6.1 より、CARMA ロード・モジュール CRASERV は、CARMA カスタム情報 VSAM データ・セットの固定長フォーマットと可変長フォーマットの両方をサポートします。
- 古いバージョンの CARMA ロード・モジュールは、可変長フォーマットをサポートしないため、可変長の CARMA カスタム情報 VSAM データ・セットとともに使用すると、ストリングの文字化けが発生します。

CARMA への RSE インターフェース

CARMA サーバーは、他のホスト・ベースの製品で 1 つ以上の Software Configuration Manager (SCM) にアクセスするための標準 API を提供します。しかし、クライアント PC と直接通信するための方式は提供しません。このため、RSE サーバーなどの他の製品に依存します。RSE サーバーは、CRASRV.properties 内の設定を使用して CARMAサーバーの始動および接続を行います。

CRASRV.properties は /etc/rdz/ に置かれます。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。このファイルは、TSO **OEDIT** コマンドで編集できます。

注: 加えた変更があれば、それを取得するために RSED 開始タスクを再始動する必要があります。

```
# CRASRV.properties - CARMA configuration options
#
port.start=5227
port.range=100
startup.script.name=/usr/lpp/rdz/bin/carma.startup.rex
clist.dsname='FEK.#CUST.CNTL(CRASUBMT)'
crastart.stub=/usr/lpp/rdz/bin/CRASTART
crastart.configuration.file=/etc/rdz/crastart.conf
crastart.syslog=Partial
crastart.timeout=420
#crastart.steplib=FEK.SFEKLPA
#crastart.tasklib=TASKLIB
```

図 10. CRASRV.properties - CARMA 構成ファイル

port.start

CARMA と RSE サーバーとの間の通信に使用される最初のポート。デフォルト・ポートは 5227 です。このポート上の通信は、ご使用のホスト・マシンだけに限定されます。

注: ポートを選択する前に、**NETSTAT** および **NETSTAT PORTL** コマンドで、そのポートがシステム上で使用可能であることを確認してください。詳しくは、161 ページの『予約済み TCP/IP ポート』を参照してください。

port.range

CARMA 通信に使用される、port.start から始まるポートの範囲。デフォルトは 100 です。例えば、デフォルトを使用した場合、CARMA はポート 5227 から 5326 (これ自体を含む) までを使用できます。

startup.script.name

CARMA 始動スクリプトの絶対パスを定義します。デフォルトは /usr/lpp/rdz/bin/carma.startup.rex です。この REXX exec は CARMA サーバーの始動をトリガーします。

clist.dsname

CARMA サーバーの始動方式を定義します。

- *CRASTART は、CRASTART を使用して、CARMA サーバーを RSE 内のサブタスクとして始動するよう指示します。詳細については、62 ページの『(オプション) CRASTART を使用した代替の CARMA サーバーの始動』を参照してください。*CRASTART を指定する場合は、crastart.* ディレクティブも指定する必要があります。
- *ISPF は、ISPF の TSO/ISPF クライアント・ゲートウェイを使用して CARMA サーバーを始動するよう指示します。詳細については、65 ページの『(オプション) TSO/ISPF クライアント・ゲートウェイを使用した代替の CARMA サーバーの始動』を参照してください。
- それ以外のすべての値は、TSO のような命名規則を使用して、CRASUBMT CLIST のロケーションを定義します。引用符 (') を付けた場合、そのデータ・セット名は絶対参照となり、引用符 (') を付けなかった場合、そのデータ・セット名には TSO 接頭部ではなく、クライアント・ユーザー ID が接頭部として付加されます。後者の場合は、すべての CARMA ユーザーが独自の CRASUBMT CLIST を保守する必要があります。

デフォルトは 'FEK.#CUST.CNTL(CRASUBMT)' です。この CLIST は、バッチ実行依頼方式で接続を開いたときに CARMA サーバーを始動します。

crastart.stub

CRASTART を呼び出すための z/OS UNIX スタブ。デフォルトは /usr/lpp/rdz/bin/CRASTART です。このスタブは、MVS ベースの CRASTART ロード・モジュールを z/OS UNIX プロセスから使用できるようにします。このディレクティブは、clist.dsname ディレクティブの値が *CRASTART である場合にのみ使用されます。

crastart.configuration.file

CRASTART 構成ファイルの名前を指定します。デフォルトは /etc/rdz/crastart.conf です。このファイルは、CARMA サーバーを始動するために必要なデータ・セット割り振りとプログラム呼び出しを定義します。このディレクティブは、clist.dsname ディレクティブの値が *CRASTART である場合にのみ使用されます。

crastart.syslog

CRASTART で CARMA サーバーを始動するときに、どれくらいの量の情報をシステム・ログに書き込むかを指定します。デフォルトは Partial です。有効な値は、以下のとおりです。

A (All)	すべてのトレース情報を SYSLOG に出力します。
P (Partial)	接続、切断、およびエラー情報のみを SYSLOG に出力します。
それ以外のすべて	エラー条件のみを SYSLOG に出力します。

このディレクティブは、clist.dsname ディレクティブの値が *CRASTART である場合にのみ使用されます。

crastart.timeout

アクティビティーがないために CARMA サーバーが終了するまでの時間の長さ (秒単位)。デフォルトは 420 (7 分) です。このディレクティブは、clist.dsname ディレクティブの値が *CRASTART である場合にのみ使用されます。

crastart.steplib

rsed.envvars 内の STEPLIB ディレクティブを通じてアクセスする場合の CRASTART モジュールのロケーション。デフォルトは FEK.SFEKLPA です。CRASTART モジュールを LPA または LINKLIST に含めることができない場合は、このディレクティブをコメント解除してカスタマイズします。CRASTART モジュールが LPA 内にない場合は、プログラム制御および APF の問題が起きる可能性があることに注意してください。このディレクティブは、clist.dsname ディレクティブの値が *CRASTART である場合にのみ使用されます。

crastart.tasklib

crastart.conf 内の TASKLIB DD 名の代替名。デフォルトは TASKLIB です。使用している SCM または RAM に DD 名 TASKLIB が特殊な意味を持ち、STEPLIB の代わりに使用できない場合は、このディレクティブをコメント解除してカスタマイズしてください。このディレクティブは、clist.dsname ディレクティブの値が *CRASTART である場合にのみ使用されます。

バッチ実行依頼を使用した CARMA サーバーの始動

このセクションでは、Developer for System z で CARMA サーバーを始動するデフォルトの方式の構成方法について説明します。別の始動方式を使用する場合は、このカスタマイズ・ステップを迂回できます。

Developer for System z は、バッチ実行依頼 CARMA サーバー始動方式をデフォルトで使用します。この方式では、CRASTART モジュールが LPA 内に存在する必要がなく、TSO/ISPF クライアント・ゲートウェイにも依存しません。この方式は、CARMA サーバーをユーザーの JES 内の長時間実行バッチ・ジョブとして実行依頼します。

CRASRV.properties の調整

RSE サーバーは、58 ページの『CARMA への RSE インターフェース』で説明されているように、/etc/rdz/CRASRV.properties 内の設定を使用して CARMA サーバーの始動および接続を行います。このファイルは、TSO **OEDIT** コマンドで編集できます。変更を有効にするには、RSE の再始動が必要であることに注意してください。

以下の例に示すように、clist.dsname ディレクティブの値を CRASUBMT CARMA サーバー始動 CLIST のデータ・セットおよびメンバー名に変更します。各種ディレクティブの詳細については、58 ページの『CARMA への RSE インターフェース』を参照してください。

```
port.start=5227
port.range=100
startup.script.name=/usr/lpp/rdz/bin/carma.startup.rex
clist.dsname='FEK.#CUST.CNTL(CRASUBMT)'
```

図 11. CRASRV.properties - バッチ実行依頼を使用した CARMA の始動

CRASUBMT の調整

以下のコード・サンプルに示すように、CRASUBMT CLIST をカスタマイズします。カスタマイズの手順については、CRASUBMT 内のドキュメンテーションを参照してください。CRASUBMT CLIST は CARMA サーバーを実行依頼します。

CRASUBMT は FEK.#CUST.CNTL に置かれます。ただし、ジョブ FEK.SFEKSAMP (FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

```
PROC 1 PORT TIMEOUT(420)
SUBMIT * END($$)
//CRA&PORT JOB CLASS=A,MSGCLASS=A,MSGLEVEL=(1,1)
//RUN      EXEC PGM=IKJEFT01,DYNAMNBR=25,REGION=1024K,TIME=NOLIMIT
//STEPLIB  DD DISP=SHR,DSN=FEK.SFEKLOAD
//*        DD DISP=SHR,DSN=FEK.#CUST.LOAD
//CRADEF   DD DISP=SHR,DSN=FEK.#CUST.CRADEF
//CRAMSG   DD DISP=SHR,DSN=FEK.#CUST.CRAMSG
//CRASTRS  DD DISP=SHR,DSN=FEK.#CUST.CRASTRS
//*CRARAM1 DD DISP=SHR,DSN=FEK.#CUST.CRARAM1
//*
//ISPPROF  DD DISP=(NEW,DELETE,DELETE),
//          SPACE=(TRK,(1,1,5)),LRECL=80,RECFM=FB,UNIT=SYSALLDA
//ISPLIB   DD DISP=SHR,DSN=ISP.SISPMENU
//ISPPLIB  DD DISP=SHR,DSN=ISP.SISPPENU
//ISPSLIB  DD DISP=SHR,DSN=ISP.SISPSENU
//ISPTLIB  DD DISP=SHR,DSN=ISP.SISPTENU
//ISPEXEC  DD DISP=SHR,DSN=ISP.SISPEXEC
//SYSPROC  DD DISP=SHR,DSN=ISP.SISPCLIB
//*
//CARMALOG DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD *
ISPSTART PGM(CRASERV) PARM(&PORT &TIMEOUT)
//*
$$
EXIT CODE(0)
```

図 12. CRASUBMT - バッチ実行依頼を使用した CARMA の始動

注:

- CARMA TSO 環境をカスタマイズするために、独自の DD ステートメントとデータ・セット連結を追加し、TSO ログオン・プロシーチャーを模倣することができます。
- オプションとして、FEK.#CUST.CNTL(CRASUBMT) CLIST 内の PROC 1 PORT TIMEOUT(420) 行を変更することにより、CARMA のタイムアウト値を変更できます。このタイムアウト値は、CARMA がクライアントからの次のコマンドを待つ秒数です。0 の値を設定すると、デフォルトのタイムアウト値になります。つまり、現行の 420 秒 (7 分) です。
- CARMA 始動プロセスの詳細は、rsecomm.log の中に示されます。rsecomm.log の詳細レベルの詳しい設定方法については、102 ページの『(オプション) RSE トレース』を参照してください。
- 変更は、更新後に始動されたすべての CARMA サーバーについて有効になります。

(オプション) CRAFT を使用した代替の CARMA サーバーの始動

このセクションでは、Developer for System z で CARMA サーバーを始動する代替の方式の構成方法について説明します。別の始動方式を使用する場合は、このカスタマイズ・ステップを迂回できます。

Developer for System z がサポートする代替 CARMA サーバー始動方式は、TSO/ISPF クライアント・ゲートウェイに依存せず、また、JES イニシエーターを使用してサーバー・ジョブを実行依頼しません。この方式は、CRAFT を使用して CARMA サーバーを RSE 内のサブタスクとして始動するもので、TSO/ISPF クライアント・ゲートウェイ・サービスによく似ています。

注: CARMA 始動プロセスの詳細は、rsecomm.log の中に示されます。

rsecomm.log の詳細レベルの詳しい設定方法については、102 ページの『(オプション) RSE トレース』を参照してください。

CRASRV.properties の調整

RSE サーバーは、58 ページの『CARMA への RSE インターフェース』で説明されているように、/etc/rdz/CRASRV.properties 内の設定を使用して CARMA サーバーの始動および接続を行います。このファイルは、TSO OEDIT コマンドで編集できます。変更を有効にするには、RSE の再始動が必要であることに注意してください。

以下の例に示すように、clist.dsname ディレクティブの値を *CRAFT に変更し、craft.* ディレクティブに正しい値を指定してください。各種ディレクティブの詳細については、58 ページの『CARMA への RSE インターフェース』を参照してください。

```
port.start=5227
port.range=100
startup.script.name=/usr/lpp/rdz/bin/carma.startup.rex
clist.dsname=*CRAFT
craft.stub=/usr/lpp/rdz/bin/CRAFT
craft.configuration.file=/etc/rdz/craft.conf
craft.syslog=Partial
craft.timeout=420
#craft.step1lib=FEK.SFEKLPA
#craft.tasklib=TASKLIB
```

図 13. CRASRV.properties - *CRAFT 代替 CARMA 始動

注: SMFPRMxx parmlib メンバー内の JWT パラメーターの設定が CRASRV.properties のタイムアウト値より小さい場合は、モジュール CRASRV のシステム異常終了 522 が発生します。これによって、CARMA の操作が影響を受けることはありません。必要な場合、サーバーが自動的に再始動されるためです。

craft.conf の調整

このカスタマイズ・ステップでは、カスタマイズした CRASUBMT (60 ページの『バッチ実行依頼を使用した CARMA サーバーの始動』を参照) の印刷出力を手元に用意し、簡単に参照できるようにしてください。印刷出力は、メンバーをカスタマイズしていない場合でも役に立ちます。

CRASTART は `crastart.conf` 内の定義を使用して、バッチ TSO および ISPF コマンドを実行するための有効な環境を作成します。Developer for System z は、その環境を使用して、CARMA サーバーが呼び出した CRASERV を実行します。

`crastart.conf` は `/etc/rdz/` に置かれます。ただし、ジョブ FEK.SFEKSAMP (FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。このファイルは、TSO **OEDIT** コマンドで編集できます。

注: 変更は、更新後に始動されたすべての CARMA サーバーについて有効になります。

以下のカスタマイズ・ステップは、下記のコード・サンプルに示した構成ファイルを調整するために必要です。

- CRASUBMT プロシーチャーの STEPLIB 連結に割り振られたデータ・セットを、`crastart.conf` 内の TASKLIB ステートメントに追加します。
- 必須 CARMA VSAM DD の CRADEF、CRAMSG、および CRSTRS の項目を作成します。(カスタマイズした) CRASUBMT プロシーチャー内で提供されているデータ・セット名を使用してください。
- (カスタマイズした) CRASUBMT プロシーチャー内にある、カスタム DD ステートメントおよび関連するデータ・セット連結を追加します。例えば、サンプルの PDS RAM を使用している場合は、CRARAM1 DD ステートメントとデータ・セット名を追加します。データ・セット名 (DISP=SHR で割り振ったもの)、SYSOUT、および DUMMY 構造を使用できることに注意してください。
- オプションとして、-COMMAND ステートメントを使用して BPXWDYN コマンドを追加します。複数の -COMMAND ステートメントがあってもかまいません。BPXWDYN を使用すると、一時データ・セットの作成、SHR 以外の処分、他のサブシステムへの割り振りなど、より複雑な割り振りを行うことができます。BPXWDYN の詳細については、「REXX および z/OS UNIX System Services の使い方」(SA88-8644) を参照してください。
- PROGRAM ステートメントを使用して、求めるプログラム起動方式を選択します。推奨される方式は、「PROGRAM=IKJEFT01 CRASERV &CRAPRM1. &CRAPRM2.」です。この方式では、APF データ・セットと非 APF データ・セットの混用を処理できる TSO 環境が得られます。他の方式については、サンプルの `crastart.conf` を参照してください。

注: `crastart.conf` 定義を複数の行にまたがって分割することはできません。

```

* crastart.conf - CARMA allocation options

TASKLIB   = FEK.SFEKLOAD
CRADEF    = FEK.#CUST.CRADEF
CRAMSG    = FEK.#CUST.CRAMSG
CRASTRS   = FEK.#CUST.CRASTRS
*CRARAM1  = FEK.#CUST.CRARAM1
*
CARMALOG  = SYSOUT(H)
SYSTSPRT  = SYSOUT(H)
SYSTSIN   = DUMMY
-COMMAND=ALLOC FI(SCRATCH) NEW DELETE DSORG(PS) RECFM(F,B) LRECL(80) UNIT(VIO)
*
PROGRAM=IKJEFT01 CRASERV &CRAPRM1. &CRAPRM2.

```

図 14. *crastart.conf* - **CRASTART* 代替 *CARMA* 始動

構成ファイルの中で、以下の変数を使用できます。

表 10. *crastart.conf* の変数

&CRAUSER.	クライアントのログオン・ユーザー ID。
&CRADATE.	Dyyyyddd 形式 (7 文字のユリウス日付) の現在日付。
&CRATIME.	Thhmmss 形式 (時分秒) の現在時刻。
&CRAPRM3. から &CRAPRM9.	<p>ユーザーが割り当てた値を持つ追加変数。これらの変数を使用するには、<i>CRASRV.properties</i> 内の <i>startup.script.name</i> によって参照される <i>CARMA</i> 始動 REXX をカスタマイズする必要があります。</p> <p>これらの変数を使用する場合は、デフォルトの始動 REXX <i>/usr/lpp/rdz/bin/carma.startup.rex</i> のコピーをカスタマイズし、<i>startup.script.name</i> がこのコピーを指すようにする必要があります。これにより、保守の際にデフォルトの REXX が更新されても作業内容が失われなくなります。</p>
システム・シンボル	<i>SYS1.PARMLIB(IEASYMxx)</i> で定義した任意のシステム・シンボル
-<DD>	前に定義した DD 名の前にダッシュ (-) を付加すると、JCL の *.ddname 逆方向参照のように機能します。オリジナルの DD は、-COMMAND ステートメントを使用して割り振る必要があります。

注: TSO 接頭部の変数は存在しません。TSO は構成ファイルが解釈される時、アクティブではないからです。

(オプション) TSO/ISPF クライアント・ゲートウェイを使用した代替の CARMA サーバーの始動

このセクションでは、Developer for System z で CARMA サーバーを始動する代替の方式の構成方法について説明します。別の始動方式を使用する場合は、このカスタマイズ・ステップを迂回できます。

Developer for System z がサポートする代替 CARMA サーバー始動方式を使用すると、CRASTART モジュールが LPA 内に存在する必要はなく、また、JES イニシエーターを使用してサーバー・ジョブを実行依頼しません。この方式は ISPF の TSO/ISPF クライアント・ゲートウェイを使用するもので、TSO コマンド・サービスにアクセスするデフォルトの方法によく似ています。

注: CARMA 始動プロセスの詳細は、rsecomm.log の中に示されます。

rsecomm.log の詳細レベルの詳しい設定方法については、102 ページの『(オプション) RSE トレース』を参照してください。

CRASRV.properties の調整

RSE サーバーは、58 ページの『CARMA への RSE インターフェース』で説明されているように、/etc/rdz/CRASRV.properties 内の設定を使用して CARMA サーバーの始動および接続を行います。このファイルは、TSO **OEDIT** コマンドで編集できます。変更を有効にするには、RSE の再始動が必要であることに注意してください。

以下の例に示すように、clist.dsname ディレクティブの値を *ISPF に変更します。各種ディレクティブの詳細については、58 ページの『CARMA への RSE インターフェース』を参照してください。

```
port.start=5227
port.range=100
startup.script.name=/usr/lpp/rdz/bin/carma.startup.rex
clist.dsname=*ISPF
```

図 15. CRASRV.properties - *ISPF 代替 CARMA 始動

ISPF.conf の調整

このカスタマイズ・ステップでは、カスタマイズした CRASUBMT (60 ページの『バッチ実行依頼を使用した CARMA サーバーの始動』を参照) の印刷出力を手元に用意し、簡単に参照できるようにしてください。印刷出力は、メンバーをカスタマイズしていない場合でも役に立ちます。

ISPF の TSO/ISPF クライアント・ゲートウェイは、ISPF.conf 内の定義を使用して、バッチ TSO および ISPF コマンドを実行するための有効な環境を作成します。Developer for System z は、その環境を使用して CARMA サーバーを実行します。

ISPF.conf は /etc/rdz/ に置かれます。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除き

ます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。このファイルは、TSO **OEDIT** コマンドで編集できます。

注: 変更は、更新後に始動されたすべての CARMA サーバーについて有効になります。

以下のカスタマイズ・ステップは、下記のコード・サンプルに示した構成ファイルを調整するために必要です。

- 必須 ISPF データ・セットの正しい名前を指定します (ただし、ISPPROF を割り振らないでください。これは動的に割り振られます)。
- システムが CRASRV1 exec を検出できるように、Developer for System z proclib の FEK.SFEKPROC を SYSPROC または SYSEXEC ステートメントに付加します。この exec は CARMA サーバーを始動します (また、それによって CRASUBMT の SYSTSIN DD を置き換えます)。
- CRASUBMT プロシーチャーの DD STEPLIB 連結を、isp1lib ステートメントに付加します。
- 必須 CARMA VSAM DD の CRADEF、CRAMSG、および CRSTRS の項目を作成します。(カスタマイズした) CRASUBMT プロシーチャー内で提供されているデータ・セット名を使用してください。
- (カスタマイズした) CRASUBMT プロシーチャー内にある、カスタム DD ステートメントおよび関連するデータ・セット連結を追加します。例えば、サンプルの PDS RAM を使用している場合は、CRARAM1 DD ステートメントとデータ・セット名を追加します。データ・セット名 (DISP=SHR で割り振ったもの) だけを使用できることに注意してください。
- オプションとして、allocexec ディレクティブをコメント解除してカスタマイズし、exec による追加の割り振りを行います。

注: SYSTSIN、SYSTSOUT、または CARMALOG の各 DD、および、インストリーム・データや SYSOUT= などの JES 構造を使用するその他の DD ステートメントを組み込まないでください。これらの項目は、データ・セットを使用するように変換する必要があります。

```
sysproc=ISP.SISPCLIB,FEK.SFEKPROC
isp1lib=FEK.SFEKLOAD
ispmlib=ISP.SISPMENU
isptlib=ISP.SISPTENU
ispplib=ISP.SISPPENU
ispslib=ISP.SISPSLIB
CRADEF =FEK.#CUST.CRADEF
CRAMSG =FEK.#CUST.CRAMSG
CRASTRS=FEK.#CUST.CRASTRS
*CRARAM1=FEK.#CUST.CRARAM1
allocjob=FEK.#CUST.CNTL(CRAISPRX)
```

図 16. ISPF.conf - *ISPF 代替 CARMA 始動

DD CARMALOG は、ISPF.conf 内でマップできない SYSOUT=* をデフォルトで参照します。また、この DD をデータ・セットに直接マップすることもできません。すべての Developer for System z ユーザーが、同じ ISPF.conf ファイルと、したがって同じデータ・セットを使用するからです。

しかし、283 ページの『第 16 章 TSO 環境のカスタマイズ』の 285 ページの『詳細設定 - 割り振り exec の使用』のセクションで説明されているように、割り振り exec を使用して、アクティブなユーザー ID に基づいたデータ・セットを作成し、割り振ることができます。DD CARMALOG をデータ・セット名 TSOPREFIX'.'USERID'.CRA.'TIMESTAMP'.CARMALOG' に割り振る例として、データ・セット FEK.#CUST.CNTL 内のサンプル・メンバー CRAISPRX を参照してください。

注:

- allocjob ディレクティブを使用する場合は、前に ISPF.conf で行った DD 定義を元に戻さないように注意してください。
- SMFPRMxx parmlib メンバー内の JWT パラメーターの設定が ISPF.conf 内の ISPF_timeout 値よりも小さい場合は、モジュール CRASERV のシステム異常終了 522 が予想されます。これによって、CARMA の操作が影響を受けることはありません。必要な場合、サーバーが自動的に再始動されるためです。

(オプション) サンプルの Repository Access Manager (RAM) のアクティブ化

Repository Access Manager (RAM) は、z/OS Software Configuration Manager (SCM) とのインターフェースとなるユーザー作成 API です。アクティブにしたいサンプルの RAM については、以下のセクションの説明に従ってください。

注: サンプルの RAM は CARMA 環境の構成をテストする目的で、また、独自の RAM を開発するための例として提供されています。提供されているサンプルの RAM を実稼働環境で使用しないでください。

提供されているサンプルの RAM およびサンプルのソース・コードの詳細については、「*Rational Developer for System z Common Access Repository Manager Developer's Guide*」(SC23-7660) を参照してください。

参照するサンプルのメンバーは、FEK.#CUST.JCL に置かれます。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

PDS RAM のアクティブ化

PDS RAM は、「リモート・システム」ビューの「MVS ファイル」>「ユーザー・データ・セット」と同様のデータ・セット・リストを提供します。PDS RAM はデフォルトで RAM ID 0 を使用します。

注: PDS RAM は、CARMA が ISPF 内で (ISPSTART を使用して) 始動されることを必要とします。

1. FEK.#CUST.JCL(CRA#VPDS) JCL をカスタマイズして実行依頼します。カスタマイズの手順については、CRA#VPDS 内のドキュメンテーションを参照してください。CRA#VPDS は PDS RAM メッセージ VSAM データ・セットの作成と事前準備を行います。
2. CRARAM1 DD ステートメントを、選択した CARMA 始動方式に追加し、PDS RAM メッセージ VSAM のデータ・セット名を指定します。

SCLM RAM のアクティブ化

SCLM RAM は、ISPF の Software Configuration Manager である SCLM への基本的な入り口となります。SCLM RAM はデフォルトで RAM ID 1 を使用します。

注: SCLM RAM は、CARMA が ISPF 内で (ISPSTART を使用して) 始動されることを必要とします。

1. FEK.#CUST.JCL(CRA#VSLM) JCL をカスタマイズして実行依頼します。カスタマイズの手順については、CRA#VSLM 内のドキュメンテーションを参照してください。CRA#VSLM は SCLM RAM メッセージ VSAM データ・セットの作成と事前準備を行います。
2. CRARAM2 DD ステートメントを、選択した CARMA 始動方式に追加し、SCLM RAM メッセージ VSAM のデータ・セット名を指定します。
3. FEK.#CUST.JCL(CRA#ASLM) JCL をカスタマイズします。カスタマイズの手順については、CRA#ASLM 内のドキュメンテーションを参照してください。CRA#ASLM は、SCLM RAM クライアントに必要なデータ・セットを割り当てます。

注: 各ユーザーは、CARMA を SCLM RAM と一緒に使用する前に、1 回だけ FEK.#CUST.JCL(CRA#ASLM) を実行依頼する必要があります。そうしなかった場合は、割り振りエラーになります。

スケルトン RAM のアクティブ化

スケルトン RAM は、ユーザーが独自の RAM を開発する際に使用できるスケルトン・フレームワークを提供します。スケルトン RAM はデフォルトで RAM ID 3 を使用します。

1. FEK.#CUST.JCL(CRA#CRAM) JCL をカスタマイズして実行依頼します。カスタマイズの手順については、CRA#CRAM 内のドキュメンテーションを参照してください。CRA#CRAM は、スケルトン RAM をコンパイルします。
2. コンパイルしたスケルトン RAM モジュール CRARAMSA を保持しているロード・ライブラリーを、選択した CARMA 始動方式の STEPLIB DD (CRASTART 方式の TASKLIB DD) に追加します。

(オプション) CA Endeavor® SCM RAM のアクティブ化

IBM® Rational® Developer for System z Interface for CA Endeavor® Software Configuration Manager は、Developer for System z クライアントが CA Endeavor® SCM に直接アクセスできるようにします。これ以降は、IBM® Rational® Developer for System z Interface for CA Endeavor® SCM を CA Endeavor® SCM RAM (Repository Access Manager) と略記します。

この資料に掲載しているサンプルの RAM とは異なり、CA Endeavor® SCM RAM は実動タイプの RAM です。両タイプの RAM を同じセットアップでアクティブ化しないでください。

重要: CA Endeavor® SCM RAM 用に提供されているセットアップ・ジョブを実行すると、アクティブな CARMA セットアップが CA Endeavor® SCM RAM のみを保持するセットアップに置き換えられます。

注: TSO/ISPF クライアント・ゲートウェイ始動方式を、CA Endeavor® SCM RAM と併用することはできません。

要件およびチェックリスト

このカスタマイズ・タスクを完了するには、セキュリティ管理者および TCP/IP 管理者の支援が必要になります。このタスクには、以下のリソースまたは特殊なカスタマイズ・タスクが必要です。

- 内部通信用の TCP/IP ポート範囲
- (オプション) ユーザーに CRA* ジョブの実行依頼を許可するセキュリティ規則
- (オプション) LPA 更新

ご使用のサイトで CA Endeavor® SCM RAM の使用を開始するには、以下のタスクを行う必要があります。特に断りがない限り、すべてのタスクは必須です。

1. CA Endeavor® SCM RAM を CARMA に対して定義する VSAM データ・セットの割り振りと事前準備を行います。詳細については、『CA Endeavor® SCM RAM の定義』を参照してください。
2. 望ましい始動方式 (バッチ実行依頼または CRASTART) を選択し、関連する構成ファイルに必要なカスタマイズを行います。詳細については、以下を参照してください。
 - 70 ページの『バッチ実行依頼を使用した CA Endeavor® SCM RAM の始動』
 - 72 ページの『CRASTART を使用した CA Endeavor® SCM RAM の始動』
3. オプションとして、ユーザー固有のデータ・セットの動的割り振りに使用される割り振り exec をカスタマイズします。詳細については、74 ページの『(オプション) CRANDVRA のカスタマイズ』を参照してください。
4. オプションとして、CA Endeavor® SCM RAM 固有の構成ファイルをカスタマイズします。詳細については、75 ページの『(オプション) CA Endeavor® SCM RAM のカスタマイズ』を参照してください。

CA Endeavor® SCM RAM の定義

以下の CARMA コンポーネントは、選択する始動方式に関係なく、必ずカスタマイズしなければなりません。参照するサンプルのメンバーは、FEK.#CUST.JCL に置かれます。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

1. FEK.#CUST.JCL(CRA#VCAD) JCL をカスタマイズして実行依頼します。カスタマイズの手順については、CRA\$VDEF 内のドキュメンテーションを参照してください。CRA#VCAD は、CARMA 構成 VSAM データ・セット CRADEF の作成と事前準備を行います。
2. FEK.#CUST.JCL(CRA\$VMSG) JCL をカスタマイズして実行依頼します。カスタマイズの手順については、CRA\$VMSG 内のドキュメンテーションを参照してください。CRA\$VMSG は、CARMA メッセージ VSAM データ・セット CRAMSG の作成と事前準備を行います。

注: これは、サンプルの RAM の場合と同じジョブです。

3. FEK.#CUST.JCL(CRA#VCAS) JCL をカスタマイズして実行依頼します。カスタマイズの手順については、CRA\$VSTR 内のドキュメンテーションを参照してください。CRA#VCAS は、CARMA カスタム情報 VSAM データ・セット CRASTRS の作成と事前準備を行います。

注:

- CA Endeavor® SCM RAM は、デフォルトで RAM ID 0 を使用します。
- (カスタムの) RAM の定義を既存の VSAM 構成にマージする必要がある場合は、サンプル・ジョブ FEK.#CUST.JCL(CRA#UADD) を参照してください。このジョブは、変更する CARMA VSAM ごとにカスタマイズし、実行依頼する必要があります。各種の CARMA VSAM が使用するレコード構造の詳細については、「*Rational Developer for System z Common Access Repository Manager Developer's Guide*」(SC23-7660) を参照してください。
- VSAM から順次データ・セットへアクティブな定義を抽出するには、サンプル・ジョブ FEK.#CUST.JCL(CRA#UQRY) を使用します。

バッチ実行依頼を使用した CA Endeavor® SCM RAM の始動

CA Endeavor® SCM RAM を使用する CARMA サーバーを CRASTART 方式で始動する場合は、このステップを実行しないでください。

Developer for System z は、バッチ実行依頼 CARMA サーバー始動方式を使用して、CA Endeavor® SCM RAM を始動することができます。この方式は、CARMA サーバーをユーザーの JES 内の長時間実行バッチ・ジョブとして実行依頼します。

バッチ実行依頼始動方式の詳細については、60 ページの『バッチ実行依頼を使用した CARMA サーバーの始動』を参照してください。

CRASRV.properties の調整

RSE サーバーは、58 ページの『CARMA への RSE インターフェース』で説明されているように、/etc/rdz/CRASRV.properties 内の設定を使用して CARMA サーバーの始動および接続を行います。このファイルは、TSO OEDIT コマンドで編集できます。変更を有効にするには、RSE の再始動が必要であることに注意してください。

以下の例に示すように、clist.dsname ディレクティブの値を CRASUBCA CARMA サーバー始動 CLIST のデータ・セットおよびメンバー名に変更します。各種ディレクティブの詳細については、58 ページの『CARMA への RSE インターフェース』を参照してください。

```
port.start=5227
port.range=100
startup.script.name=/usr/lpp/rdz/bin/carma.startup.rex
clist.dsname='FEK.#CUST.CNTL(CRASUBCA)'
```

図 17. 図 x1. CRASRV.properties - バッチ実行依頼を使用した CA Endeavor® SCM RAM の始動

CRASUBCA の調整

以下のコード・サンプルに示すように、CRASUBCA CLIST をカスタマイズします。カスタマイズの手順については、CRASUBCA 内のドキュメンテーションを参照してください。CRASUBCA CLIST は、CA Endeavor® SCM 用に CARMA サーバーを実行依頼します。

CRASUBCA は FEK.#CUST.CNTL に置かれます。ただし、ジョブ FEK.SFEKSAMP (FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

```
PROC 1 PORT TIMEOUT(420)
SUBMIT * END($$)
//CRA&PORT JOB CLASS=A,MSGCLASS=A,MSGLEVEL=(1,1)
//RUN      EXEC PGM=IKJEFT01,DYNAMNBR=125,REGION=0M,TIME=NOLIMIT,
//          PARM='%CRANDVRA NDVRC1 PGM(CRASERV) PARM(&PORT &TIMEOUT)'
//STEPLIB DD DISP=SHR,DSN=FEK.SFEKLOAD
//          DD DISP=SHR,DSN=CA.NDVR.AUTHLIB
//          DD DISP=SHR,DSN=CA.NDVRU.AUTHLIB
//CRADEF DD DISP=SHR,DSN=FEK.#CUST.CRADEF
//CRAMSG DD DISP=SHR,DSN=FEK.#CUST.CRAMSG
//CRASTRS DD DISP=SHR,DSN=FEK.#CUST.CRASTRS
//*
//SYSPROC DD DISP=SHR,DSN=ISP.SISPCLIB
//          DD DISP=SHR,DSN=FEK.SFEKPROC
//ISPEXEC DD DISP=SHR,DSN=ISP.SISPEXEC
//ISPMLIB DD DISP=SHR,DSN=ISP.SISPMENU
//ISPPLIB DD DISP=SHR,DSN=ISP.SISPPENU
//ISPSLIB DD DISP=SHR,DSN=ISP.SISPSENU
//ISPTLIB DD DISP=SHR,DSN=ISP.SISPTENU
//ISPCTL0 DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
//          SPACE=(TRK,(1,1)),LRECL=80,RECFM=FB
//ISPCTL1 DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
//          SPACE=(TRK,(1,1)),LRECL=80,RECFM=FB
//ISPPROF DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
//          SPACE=(TRK,(1,1,5)),LRECL=80,RECFM=FB
//*
//CARMALOG DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD DUMMY
//*
//CONLIB DD DISP=SHR,DSN=CA.NDVR.CONLIB
//JCLOUT DD SYSOUT=(A,INTRDR),DCB=(LRECL=80,RECFM=F,BLKSIZE=80)
//EXT1ELM DD DISP=(NEW,DELETE),UNIT=SYSALLDA,
//          RECFM=VB,LRECL=4096,BLKSIZE=27998,SPACE=(TRK,(5,5))
//EXT1DEP DD DISP=(NEW,DELETE),UNIT=SYSALLDA,
//          RECFM=VB,LRECL=4096,BLKSIZE=27998,SPACE=(TRK,(5,5))
//MSG3FILE DD DISP=(NEW,DELETE),UNIT=SYSALLDA,
//          RECFM=FB,LRECL=133,BLKSIZE=27930,SPACE=(TRK,(5,5))
//C1MSGSGS1 DD DISP=(NEW,DELETE),UNIT=SYSALLDA,
//          RECFM=FB,LRECL=133,BLKSIZE=27930,SPACE=(TRK,(5,5))
//C1EXMSGSGS DD DISP=(NEW,DELETE),UNIT=SYSALLDA,
//          RECFM=FB,LRECL=133,BLKSIZE=27930,SPACE=(TRK,(5,5))
//TYPEMAP DD DISP=SHR,DSN=FEK.#CUST.PARMLIB(CRATMAP)
//SHOWVIEW DD DISP=SHR,DSN=FEK.#CUST.PARMLIB(CRASHOW)
$$
EXIT CODE(0)
```

図 18. 図 x2. CRASUBCA - パッチ実行依頼を使用した CA Endeavor® SCM RAM の始動

注:

- CARMA TSO 環境をカスタマイズするために、独自の DD ステートメントとデータ・セット連結を追加し、TSO ログオン・プロシーチャーを模倣することができます。
- オプションとして、CLIST 内の PROC 1 PORT TIMEOUT(420) 行を変更することにより、CARMA のタイムアウト値を変更できます。このタイムアウト値は、CARMA がクライアントからの次のコマンドを待つ秒数です。0 の値を設定すると、デフォルトのタイムアウト値になります。つまり、現行の 420 秒 (7 分) です。
- CARMA 始動プロセスの詳細は、rsecomm.log の中に示されます。rsecomm.log の詳細レベルの詳しい設定方法については、102 ページの『(オプション) RSE トレース』を参照してください。
- 変更は、更新後に始動されたすべての CARMA サーバーについて有効になります。

CRASTART を使用した CA Endevor® SCM RAM の始動

CA Endevor® SCM RAM を使用する CARMA サーバーをバッチ実行依頼方式で始動する場合は、このステップを実行しないでください。

Developer for System z は、CRASTART CARMA サーバー始動方式を使用して、CA Endevor® SCM RAM を始動することができます。この方式は、CRASTART を使用して CARMA サーバーを RSE 内のサブタスクとして始動します。

CRASTART 始動方式の詳細については、62 ページの『(オプション) CRASTART を使用した代替の CARMA サーバーの始動』を参照してください。

注: CARMA 始動プロセスの詳細は、rsecomm.log の中に示されます。rsecomm.log の詳細レベルの詳しい設定方法については、102 ページの『(オプション) RSE トレース』を参照してください。

CRASRV.properties の調整

RSE サーバーは、58 ページの『CARMA への RSE インターフェース』で説明されているように、/etc/rdz/CRASRV.properties 内の設定を使用して CARMA サーバーの始動および接続を行います。このファイルは、TSO OEDIT コマンドで編集できます。変更を有効にするには、RSE の再始動が必要であることに注意してください。

以下の例に示すように、clist.dsname ディレクティブの値を *CRASTART に変更し、crastart.* ディレクティブに正しい値を指定してください。各種ディレクティブの詳細については、58 ページの『CARMA への RSE インターフェース』を参照してください。


```

| port.start=5227
| port.range=100
| startup.script.name=/usr/lpp/rdz/bin/carma.startup.rex
| clist.dsname=*CRASTART
| crastart.stub=/usr/lpp/rdz/bin/CRASTART
| crastart.configuration.file=/etc/rdz/crastart.endevor.conf
| crastart.syslog=Partial
| crastart.timeout=420
| #crastart.steplib=FEK.SFEKLPA
| #crastart.tasklib=TASKLIB

```

図 19. 図 x3. CRASRV.properties - CRASTART を使用した CA Endeavor® SCM RAM の始動

注: SMFPRMxx parmlib メンバー内の JWT パラメーターの設定が CRASRV.properties のタイムアウト値より小さい場合は、モジュール CRASERV のシステム異常終了 522 が発生します。これによって、CARMA の操作が影響を受けることはありません。必要であれば、サーバーが自動的に再始動されるためです。

crastart.endevor.conf の調整

CRASTART は crastart.endevor.conf 内の定義を使用して、CA Endeavor® SCM を呼び出すための有効な (TSO/ISPF) 環境を作成します。Developer for System z は、この環境を使用して CA Endeavor® SCM RAM を実行します。

crastart.endevor.conf は /etc/rdz/ に置かれます。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。このファイルは、TSO OEDIT コマンドで編集できます。

注: 変更は、更新後に始動されたすべての CARMA サーバーについて有効になります。

図 20. *crastart.conf* - CRASTART を使用した CA Endeavor® SCM RAM の始動

バッチ実行依頼始動方式と CRAFTSTART 始動方式はどちらも、REXX exec CRANDVRA を呼び出して、CA Endeavor® SCM RAM が使用するユーザー固有のデータ・セットを割り振ります。

74 IBM Rational Developer for System z: ホスト構成ガイド

特定のデフォルト (データ・セット名など) がご使用のサイトの標準に適合しない場合は、この割り振り REXX exec のコピーをカスタマイズできます。CRANDVRA は FEK.SFEKPROC に置かれます。ただし、Developer for System z の SMP/E インストール時に別の高位修飾子を使用した場合は除きます。

カスタマイズの手順については、CRANDVRA 内のドキュメンテーションを参照してください。

注: サンプルの割り振り REXX を新しいデータ・セットにコピーし、そのコピーをカスタマイズして、保守の適用時に上書きされないようにしてください。これを行う場合は、選択した CARMA 始動方式の SYSEXEC DD で SFEKPROC への参照を更新してご使用の新しいデータ・セット名に合わせる必要があります。

(オプション) CA Endevor® SCM RAM のカスタマイズ

以下の CARMA コンポーネントは、選択した始動方式に関係なく、カスタマイズすることができます。参照するサンプルのメンバーは、FEK.#CUST.PARMLIB に置かれます。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

1. (オプション) FEK.#CUST.PARMLIB(CRASHOW) をカスタマイズします。カスタマイズの手順については、CRASHOW 内のドキュメンテーションを参照してください。CRASHOW は、CA Endevor® SCM の環境やシステムなどに対するデフォルト・フィルターを定義します。
2. (オプション) FEK.#CUST.PARMLIB(CRATMAP) をカスタマイズします。カスタマイズの手順については、CRATMAP 内のドキュメンテーションを参照してください。CRATMAP は、CA Endevor® SCM タイプとファイル拡張子のマッピングをオーバーライドします。

(オプション) 複数の RAM のサポート

CARMA では、複数の RAM を定義し、それらを同時に実行することができます。ただし、1 ユーザーにつきアクティブな CARMA サーバーは 1 つのみであるため、複数の RAM がある場合でも、そのセットアップを機能させるために構成の変更が必要となる場合があります。

RAM は、RAM 開発者によって CARMA 構成 VSAM データ・セット CRADEF 内で定義されます。CARMA サーバー CRASERV は、始動時に、定義されているすべての RAM を識別し、その情報を CARMA クライアントに渡します。これでユーザーは 1 つ以上の RAM を選択できるようになり、それらが CARMA サーバーにロードされます。

RAM は CARMA サーバーのプラグインとしてアクティブになるため、各 RAM のすべての前提条件 (データ・セット割り振りなど) が CARMA サーバーのアドレス・スペースで使用可能であることを確認する必要があります。この場合、Developer for System z とともに出荷される CARMA 構成サンプル (CRASUBMT や crastart.conf など) の変更が必要になる場合があります。

例

以下の例では、CA Endeavor[®] SCM RAM を使用する既存のセットアップから
CRASTART 始動方式を使用して始動し、サンプルの PDS RAM を追加します。

CA Endeavor[®] SCM RAM 用の定義:

- FEK.SFEKVSM2(CRA0VCAD) - CRADEF 定義
- FEK.SFEKVSM2(CRA0VCAS) - CRASTRS 定義
- /etc/rdz/crastart.endeavor.conf - CRASTART 構成ファイル

PDS RAM 用の定義:

- FEK.SFEKVSM2(CRA0VDEF) - CRADEF 定義
- FEK.SFEKVSM2(CRA0VSTR) - CRASTRS 定義
- FEK.#CUST.CRARAM1 - CRARAM1 定義

このプロセスは、システム・プログラマーがセットアップを完了するために必要と
するデータと情報を、RAM 開発者が収集することから始まります。

1. PDS RAM に固有のデータを SFEKVSM2 メンバーから抽出します (これらのメン
バーは、PDS RAM だけでなくすべてのサンプル RAM の定義を保持していま
す)。
2. このデータを CA Endeavor[®] SCM RAM SFEKVSM2 メンバーとマージします。
3. PDS RAM 固有の前提条件のリストを作成します。
 - FEK.#CUST.CRARAM1 にリンクされた DD CRARAM1
 - TSO 環境

次に、システム・プログラマーがこのデータを使用して更新された CARMA VSAM
データ・セットを作成し、前提条件情報を使用して、両方の RAM をサポートでき
る CRASTART 構成ファイルを作成します。

1. この結合データを、CRA\$VDEF および CRA\$VSTR ジョブへの入力として使用し
て、更新された CARMA 構成および CARMA カスタム情報 VSAM データ・セ
ット (CRADEF と CRASTRS) を作成します。
2. CRARAM1 定義を crastart.endeavor.conf に追加します。

```
CRARAM1 = FEK.#CUST.CRARAM1
```

3. crastart.endeavor.conf 内の PROGRAM ステートメントを検証して、両方の
RAM に必要な環境を提供できることを確認します。

```
PROGRAM=IKJEFT01 %CRANDVRA NDVRC1 PGM(CRASERV)  
PARM(&CRAPRM1. &CRAPRM2.)
```

- IKJEFT01: 無許可の環境で特定の許可された呼び出しを可能にするために使用
され、また、CA Endeavor[®] SCM RAM 事前割り振り exec を実行する環境と
して使用される TSO。
- %CRANDVRA: 一時的 (および永続的) なユーザー固有の作業データ・セットを割
り振る CA Endeavor[®] SCM RAM 事前割り振り exec (FEK.SFEKPROC に置かれ
ます)。
- NDVRC1: TSO コマンドと ISPF コマンドを実行するメカニズムが組み込まれ
た、CA Endeavor[®] バックエンド。

- PGM(CRASERV): CARMA サーバーを始動する、ISPF コマンド・フォーマットの
コマンド
- PARM(&CRAPRM1. &CRAPRM2.): ISPF コマンド・フォーマットの CRASERV の
パラメーター。&CRAPRM1 は使用されるポート、&CRAPRM2 はタイムアウト値で
す。

CA Endeavor® SCM RAM は ISPF 環境でアクティブになります。これは、PDS
RAM に必要な TSO 環境も使用可能であることを意味します。

(オプション) IRXJCL と CRAXJCL

TSO (IKJEFTxx) を使用して CARMA サーバーを始動した場合、使用した RAM が
サービスを呼び出し、さらにそのサービスが IRXJCL REXX バッチ・インターフェ
ースを呼び出すと、問題が発生することがあります。問題が発生する可能性がある
のは、RAM によって呼び出されたプロセッサが、以前は TSO なしで稼働してい
たかオンライン TSO 内でのみ稼働しており、それが DD SYSTSIN または
SYSTSPRT を動的に割り振ったときです。この問題を回避するために、サンプル・
プログラム CRAXJCL が提供されています。

プロセッサは、SYSTSIN または SYSTSPRT (IRXJCL に必須) を割り振ろうとし
た場合に失敗することがあり、これは、バッチ TSO (CARMA に必須) がすでにそ
れらの DD 名を割り振って開いているためです。CRAXJCL 置換モジュールは
SYSTSIN および SYSTSPRT を DUMMY に割り振ろうとしますが、割り振りが失
敗した場合に発生するエラーを無視します。

このことは、TSO によって始動された CARMA 環境内でプロセッサを稼働した
場合、SYSTSIN および SYSTSPRT への割り振りは CARMA によって使用される
割り振りと同じものであることを意味します。TSO/CARMA の外部でプロセッサ
を稼働した場合、SYSTSIN および SYSTSPRT の割り振りは CRAXJCL によっ
て作成されます。したがって、使用するプロセッサは、SYSTSIN へ割り振られた
データ・セットの内容に依存してはなりません。

IRXJCL の呼び出しは、「TSO/E REXX 解説書」(SA88-8635) で説明されているよう
に、PARM フィールドを使用して REXX 名および始動パラメーターを渡すものと
想定されています。これは、SYSTSIN が CARMA で安全に使用できることを意味
します。IRXJCL によって SYSTSPRT へ送られたすべての出力は、最終的に
CARMA のログに収められます。

CRAXJCL 置換モジュールを呼び出すプロセッサは、CRAXJCL を呼び出す前に
DD SYSTSIN または SYSTSPRT の割り振りを試みてはなりません。

CRAXJCL の作成

CRAXJCL 置換モジュールはソース形式で出荷されます。これは、お客様がこのモ
ジュールをカスタマイズして、SYSTSPRT に使用する固有の割り振りを指定する必
要があるためです。SYSTSIN は、通常の場合、ダミー・データ・セットに割り振っ
てください。

サンプルのアセンブラー・ソース・コードおよびサンプルのコンパイル/バインド・
ジョブは、それぞれ FEK.#CUST.ASM(CRAXJCL) および FEK.#CUST.JCL(CRA#CIRX) と

して出荷されます。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

メンバー内のドキュメンテーションを使用し、必要に応じて CRAXJCL アセンブラー・ソース・コードをカスタマイズしてください。後で、CRA#CIRX JCL をカスタマイズおよび実行依頼して、CRAXJCL ロード・モジュールを作成します。カスタマイズの手順については、CRA#CIRX 内のドキュメンテーションを参照してください。

第 4 章 (オプション) Application Deployment Manager

Developer for System z は、さまざまなコンポーネントについて共通するデプロイメントの方法として、Application Deployment Manager の特定の機能を使用します。この章に示すカスタマイズ・ステップは、開発者が以下のいずれかの機能を使用する場合に必要です。

- エンタープライズ・サービス・ツール (EST)
- BMS Screen Designer
- MFS Screen Designer
- CICSTS コード生成

注: エンタープライズ・サービス・ツール (EST) には、サービス・フロー・モデラー (SFM) や エンタープライズ用 XML サービス (XSE) などの複数のツールが含まれます。

Application Deployment Manager のカスタマイズでは、CICS リソース定義 (CRD) サーバーが追加されます。このサーバーは、z/OS 上で CICS アプリケーションとして実行されて、以下の機能をサポートします。

- CICS リソース照会
- CICSplex SM 環境および非 CICSplex SM 環境での CICS リソース定義のインストールとアンインストールの要求
- プログラムおよびマップ・セットの段階的導入要求
- パイプライン・スキャン要求
- マニフェストのエクスポート、インポート、および更新要求

CICS 管理者は、269 ページの『第 15 章 CICSTS に関する考慮事項』で CRD サーバーの詳細を知ることができます。

要件およびチェックリスト

このカスタマイズ・タスクを完了するには、CICS 管理者、TCP/IP 管理者、およびセキュリティ管理者の支援が必要になります。このタスクには、以下のリソースまたは特殊なカスタマイズ・タスクが必要です。

- 外部通信用の TCP/IP ポート
- CICS 領域 JCL を更新する
- CICS 領域 CSD を更新する
- CICS 領域に対してグループを定義する
- 管理者に Application Deployment Manager VSAM の更新を許可するセキュリティ規則
- CICSTS セキュリティー・セットアップ
- (オプション) CICS トランザクション名を定義する

- (オプション) ユーザーに Application Deployment Manager VSAM の更新を許可するセキュリティ規則

ご使用のサイトで Application Deployment Manager の使用を開始するには、以下のタスクを行う必要があります。特に断りがない限り、すべてのタスクは必須です。

1. CRD リポジトリを作成します。詳細については、『CRD リポジトリ』を参照してください。
2. 使用する CICS インターフェース (RESTful または Web サービス) を選択します (両インターフェースは共存可能です)。詳細については、81 ページの『RESTful と Web サービス』を参照してください。
3. 必要に応じて、RESTful 固有のカスタマイズを行います。詳細については、82 ページの『RESTful インターフェースを使用する CRD サーバー』を参照してください。
 - CICS 主接続領域に対して CRD サーバーを定義します。
 - オプションとして、CICS 非主接続領域に対して CRD サーバーを定義します。
 - オプションとして、CRD サーバー・トランザクション ID をカスタマイズします。
4. 必要に応じて、Web サービス固有のカスタマイズを行います。詳細については、83 ページの『Web サービス・インターフェースを使用する CRD サーバー』を参照してください。
 - パイプライン・メッセージ・ハンドラーを (場合によってはカスタマイズして) CICS RPL 連結に追加します。
 - CICS 主接続領域に対して CRD サーバーを定義します。
 - オプションとして、CICS 非主接続領域に対して CRD サーバーを定義します。
5. オプションとして、マニフェスト・リポジトリを作成します。詳細については、86 ページの『(オプション) マニフェスト・リポジトリ』を参照してください。

CRD リポジトリ

ジョブ ADNVCRD をカスタマイズおよび実行依頼して、CRD リポジトリ VSAM データ・セットの割り振りと初期化を行います。カスタマイズの手順については、メンバー内のドキュメンテーションを参照してください。

ADNVCRD は FEK.#CUST.JCL にあります。ただし、ジョブ FEK.SFEKSAMP (FEKSETUP) をカスタマイズして実行依頼したときに別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

CICS 主接続領域ごとに別々のリポジトリを作成する必要があります。リポジトリの共用は、関連するすべての CICS 領域がリポジトリ内に保管された同じ値を使用することを意味します。

注:

- 既存の CRD サーバー・リポジトリを拡張して、Developer for System z バージョン 7.6.1 の管理ユーティリティに追加された URIMAP サポートを使用できるようにする必要があります。詳細については、277 ページの『管理ユーティリティのマイグレーションに関する注』を参照してください。
- 特に断りが無い限り、現行の (カスタマイズされた値を保持している) CRD サーバー・リポジトリを複数の Developer for System z リリースにまたがって再利用できます。

ユーザーには CRD リポジトリへの READ アクセス権が必要で、CICS 管理者には UPDATE アクセス権が必要です。

CICS 管理ユーティリティ

Developer for System z が提供する管理ユーティリティを使用して、CICS 管理者は CICS リソース定義のデフォルト値を指定できます。これらのデフォルトは、読み取り専用とするか、アプリケーション開発者による編集を可能にすることができます。

管理ユーティリティは、サンプル・ジョブ ADNJSAPU によって呼び出されます。このユーティリティを使用するには、CRD リポジトリに対する UPDATE アクセス権が必要です。

ADNJSAPU は FEK.#CUST.JCL に置かれます。ただし、ジョブ FEK.SFEKSAMP (FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

詳細は、269 ページの『第 15 章 CICSSTS に関する考慮事項』に説明があります。

RESTful と Web サービス

CICS Transaction Server バージョン 4.1 以上では、Representational State Transfer (RESTful) の原則に従って設計された HTTP インターフェースをサポートしています。現在この RESTful インターフェースは、戦略的な CICSSTS インターフェースとしてクライアント・アプリケーションで使用されています。従来の Web サービス・インターフェースはすでに安定化しており、今後は RESTful インターフェースのみが機能拡張の対象となります。

Application Deployment Manager は、この指示書に従い、Developer for System z バージョン 7.6 以上で新たに導入されたすべてのサービスに RESTful CRD サーバーを必要とします。

必要であれば、1 つの CICS 領域で RESTful インターフェースと Web サービス・インターフェースを同時にアクティブにすることができます。この場合、その領域で 2 つの CRD サーバーがアクティブになります。両サーバーは、同じ CRD リポジトリを共有します。2 番目のインターフェースを領域に対して定義すると、CICS から定義の重複に関する警告が発行されるので注意してください。

RESTful インターフェースを使用する CRD サーバー

このセクションでは、RESTful インターフェースを使用して Developer for System z クライアントと通信する CRD サーバーの定義方法について説明します。

必要であれば、1 つの CICS 領域で RESTful インターフェースと Web サービス・インターフェースを同時にアクティブにすることができます。この場合、その領域で 2 つの CRD サーバーがアクティブになります。両サーバーは、同じ CRD リポジトリを共有します。2 番目のインターフェースを領域に対して定義すると、CICS から定義の重複に関する警告が発行されるので注意してください。

CICS 主接続領域

CRD サーバーを主接続領域に対して定義する必要があります。主接続領域は、Developer for System z からの Web サービス要求を処理する Web Owning Region (WOR) です。

- ロード・モジュール FEK.SFEKLOAD(ADNCRD*、ADNANAL、および ADNREST) を、CICS 主接続領域の CICS RPL 連結 (DD ステートメント DFHRPL) の中に置いてください。この作業は、適用された保守が自動的に CICS で使用可能になるように、インストール・データ・セットを連結に追加する方法で行うことをお勧めします。
- ジョブ ADNCSDRS をカスタマイズおよび実行依頼して、CICS 主接続領域の CICS システム定義 (CSD) を更新します。カスタマイズの手順については、メンバー内のドキュメンテーションを参照してください。

ADNCSDRS は FEK.#CUST.JCL に置かれます。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

- 適切な CEDA コマンドを使用して、この領域の Application Deployment Manager グループをインストールします。例えば、次のようにします。

```
CEDA INSTALL GROUP(ADNPCRGP)
```

CICS 非主接続領域

CRD サーバーを 1 つ以上の追加の非主接続領域にも使用できます。それらの領域は通常、Application Owning Regions (AOR) です。

注: CICS リソース定義の管理に CICSplex® SM ビジネス・アプリケーション・サービス (BAS) を使用している場合には、これらのステップを実行する必要はありません。

- Application Deployment Manager ロード・モジュール FEK.SFEKLOAD(ADNCRD*) を、これらの非主接続領域の CICS RPL 連結 (DD ステートメント DFHRPL) の中に置きます。この作業は、適用された保守が自動的に CICS で使用可能になるように、インストール・データ・セットを連結に追加する方法で行うことをお勧めします。
- ジョブ ADNCSRAR をカスタマイズおよび実行依頼して、これらの非主接続領域の CSD を更新します。カスタマイズの手順については、メンバー内のドキュメンテーションを参照してください。

ADNCSDAR は FEK.#CUST.JCL にあります。ただし、ジョブ FEK.SFEKSAMP (FEKSETUP) をカスタマイズして実行依頼したときに別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

- 適切な CEDA コマンドを使用して、この領域の Application Deployment Manager グループをインストールします。例えば、次のようにします。

CEDA INSTALL GROUP(ADNARRGP)

(オプション) CRD サーバー・トランザクション ID のカスタマイズ

Developer for System z は、CICS リソースの定義時および照会時に、CRD サーバーが使用する複数のトランザクションを提供します。

表 11. デフォルトの CRD サーバー・トランザクション ID

トランザクション	説明
ADMS	マニフェスト処理ツールからの CICS リソース変更要求用。一般に、これは CICS 管理者が使用するためのものです。
ADMI	CICS リソースを定義、インストール、またはアンインストールする要求用。
ADMR	CICS の環境情報またはリソース情報を取り出す、上記以外のすべての要求用。

トランザクション ID をサイトの標準に合わせて変更することができます。そのためには、以下のステップを実行します。

- ADNTXNC をカスタマイズおよび実行依頼して、ロード・モジュール ADNRCUST を作成します。カスタマイズの手順については、メンバー内のドキュメンテーションを参照してください。
- 結果の ADNRCUST ロード・モジュールを、CRD サーバーが定義されている CICS 領域の CICS RPL 連結 (DD ステートメント DFHRPL) 内に配置します。
- ADNCSDTX をカスタマイズおよび実行依頼して、CRD サーバーが定義されている CICS 領域に対して ADNRCUST をプログラムとして定義します。カスタマイズの手順については、メンバー内のドキュメンテーションを参照してください。

注: RESTful CRD サーバーは、常に ADNRCUST ロード・モジュールをロードしようとします。したがって、トランザクション ID を変更しない場合でも、ADNRCUST ロード・モジュールを作成して定義しておく、パフォーマンスの面で多少のメリットがあります。

Web サービス・インターフェースを使用する CRD サーバー

このセクションでは、Web サービス・インターフェースを使用して Developer for System z クライアントと通信する CRD サーバーの定義方法について説明します。

必要であれば、1 つの CICS 領域で RESTful インターフェースと Web サービス・インターフェースを同時にアクティブにすることができます。この場合、その領域で 2 つの CRD サーバーがアクティブになります。両サーバーは、同じ CRD リポ

ジトリを共有します。2 番目のインターフェースを領域に対して定義すると、CICS から定義の重複に関する警告が発行されるので注意してください。

パイプライン・メッセージ・ハンドラー

パイプライン・メッセージ・ハンドラー (ADNTMSGH) は、SOAP ヘッダー内のユーザー ID とパスワードを処理することにより、セキュリティのために使用されます。ADNTMSGH は、サンプルのパイプライン構成ファイルによって参照されるため、CICS RPL 連結の中に入れる必要があります。パイプライン・メッセージ・ハンドラーと必要なセキュリティ・セットアップの詳細については、269 ページの『第 15 章 CICS に関する考慮事項』を参照してください。

Developer for System z は、CICS リソースの定義および照会時に、CRD サーバーが使用する複数のトランザクションを提供します。これらのトランザクション ID は、要求された操作に応じて ADNTMSGH が設定します。サイト固有の ADNTMSGH のカスタマイズができるように、サンプルの COBOL ソース・コードが提供されています。

表 12. デフォルトの CRD サーバー・トランザクション ID

トランザクション	説明
ADMS	マニフェスト処理ツールからの CICS リソース変更要求用。一般に、これは CICS 管理者が使用するためのものです。
ADMI	CICS リソースを定義、インストール、またはアンインストールする要求用。
ADMR	CICS の環境情報またはリソース情報を取り出す、上記以外のすべての要求用。

デフォルトの使用:

- FEK.SFEKLOAD(ADNTMSGH) ロード・モジュールを CICS 主接続領域の CICS RPL 連結 (DD ステートメント DFHRPL) の中に置きます。この作業は、適用された保守が自動的に CICS で使用可能になるように、インストール・データ・セットを連結に追加する方法で行うことをお勧めします。

ADNTMSGH のカスタマイズ:

サンプルのメンバー ADNMSGH* は、FEK.#CUST.JCL および FEK.#CUST.COBOL に置かれます。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

- サンプルのパイプライン・メッセージ・ハンドラー (COBOL) ソース・コード FEK.#CUST.COBOL(ADNMSGHS) を、ご使用のサイトの標準に合わせてカスタマイズします。
- ジョブ FEK.#CUST.JCL(ADNMSGHC) をカスタマイズおよび実行依頼して、カスタマイズされた ADNMSGHS ソースをコンパイルします。カスタマイズの手順については、ADNMSGHC 内のドキュメンテーションを参照してください。結果のロード・モジュールの名前は、ADNTMSGH とする必要があることに注意してください。
- 結果の ADNTMSGH ロード・モジュールを CICS 主接続領域の CICS RPL 連結 (DD ステートメント DFHRPL) の中に置きます。

注: 必ず、カスタマイズした ADNTMSGH ロード・モジュールを FEK.SFEKLOAD への参照の前に配置してください。そうしなかった場合は、デフォルトのロード・モジュールが使用されます。

CICS 主接続領域

CRD サーバーを主接続領域に対して定義する必要があります。主接続領域は、Developer for System z からのサービス要求を処理する領域です。

- ロード・モジュール FEK.SFEKLOAD(ADNCRD*、ADNANAL、および ADNREST) を、CICS 主接続領域の CICS RPL 連結 (DD ステートメント DFHRPL) の中に置いてください。これを行うためにインストール・データ・セットを連結に追加して、CICS が適用メンテナンスを自動的に使用できるようにすることを推奨します。84 ページの『パイプライン・メッセージ・ハンドラー』で説明しているように、パイプライン・メッセージ・ハンドラー・ロード・モジュール ADNTMSGH も RPL 連結の中に置く必要があることに注意してください。
- ジョブ ADNCSDWS をカスタマイズおよび実行依頼して、CICS 主接続領域の CICS システム定義 (CSD) を更新します。カスタマイズの手順については、メンバー内のドキュメンテーションを参照してください。このジョブで使用されるトランザクション ID は、パイプライン・メッセージ・ハンドラー (カスタマイズされている場合があります) が使用するトランザクション ID と一致していなければならないことに注意してください。

ADNCSDWS は FEK.#CUST.JCL に置かれます。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

- 適切な CEDA コマンドを使用して、この領域の Application Deployment Manager グループをインストールします。例えば、次のようにします。

CEDA INSTALL GROUP(ADNPCRGP)

CICS 非主接続領域

CRD サーバーを 1 つ以上の追加の非主接続領域にも使用できます。それらの領域は通常、Application Owning Regions (AOR) です。

注: CICS リソース定義の管理に CICSplex SM ビジネス・アプリケーション・サービス (BAS) を使用している場合は、以下のステップを実行する必要はありません。

- Application Deployment Manager ロード・モジュール FEK.SFEKLOAD(ADNCRD*) を、これらの非主接続領域の CICS RPL 連結 (DD ステートメント DFHRPL) の中に置きます。この作業は、適用された保守が自動的に CICS で使用可能になるように、インストール・データ・セットを連結に追加する方法で行ってください。
- ジョブ ADNCSDAR をカスタマイズおよび実行依頼して、これらの非主接続領域の CSD を更新します。カスタマイズの手順については、メンバー内のドキュメンテーションを参照してください。

ADNCSDAR は FEK.#CUST.JCL にあります。ただし、ジョブ FEK.SFEKSAMP (FEKSETUP) をカスタマイズして実行依頼したときに別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

- 適切な CEDA コマンドを使用して、この領域の Application Deployment Manager グループをインストールします。例えば、次のようにします。

CEDA INSTALL GROUP(ADNARRGP)

(オプション) マニフェスト・リポジトリ

Developer for System z では、クライアントは選択した CICS リソースを記述しているマニフェストを参照でき、オプションとして、変更することもできます。CICS 管理者によって設定された許可に応じて、変更を直接行うか、あとで CICS 管理者が処理できるよう、マニフェスト・リポジトリにエクスポートすることができます。

注:

- このステップが必要になるのは、マニフェスト処理ツールで処理するマニフェストを Developer for System z からエクスポートするお客様のみです。
- マニフェスト処理ツールは、IBM CICS エクスプローラー用のプラグインです。

ジョブ ADNVMFST をカスタマイズおよび実行依頼して、マニフェスト・リポジトリ VSAM データ・セットの割り振りと初期化を行い、それを CICS 主接続領域に対して定義します。カスタマイズの手順については、メンバー内のドキュメンテーションを参照してください。CICS 主接続領域ごとに別々のマニフェスト・リポジトリを作成する必要があります。すべてのユーザーにマニフェスト・リポジトリへの UPDATE アクセス権が必要です。

ADNVMFST は FEK.#CUST.JCL にあります。ただし、ジョブ FEK.SFEKSAMP (FEKSETUP) をカスタマイズして実行依頼したときに別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

第 5 章 (オプション) SCLM Developer Toolkit

SCLM Developer Toolkit は、SCLM の機能を拡張するために必要なツールをクライアントに提供します。SCLM 自体はホスト・ベースのソース・コード・マネージャーであり、ISPF の一部として出荷されています。

SCLM Developer Toolkit は、Eclipse ベースのプラグインを備えており、SCLM へのインターフェースになります。また、レガシー・コード開発のすべての SCLM プロセスへのアクセスを提供するほか、メインフレーム上の SCLM と同期したワークステーション上での完全な Java および J2EE 開発 (メインフレームからの J2EE コードのビルド、アセンブル、およびデプロイメントを含む) もサポートします。

要件およびチェックリスト

このカスタマイズ・タスクを完了するには、SCLM 管理者、およびオプションとしてセキュリティ管理者の支援が必要になります。このタスクには、以下のリソースか特殊なカスタマイズ・タスク、またはその両方が必要です。

- APF および LINKLIST の更新
- JAVA/J2EE サポート用の SCLM 言語変換プログラムを定義する
- JAVA/J2EE サポート用の SCLM タイプを定義する
- (オプション) ユーザーに SCLM VSAM の更新を許可するセキュリティ規則
- (オプション) Ant のインストール

ご使用のサイトで SCLM Developer Toolkit の使用を開始するには、以下のタスクを行う必要があります。特に断りがない限り、すべてのタスクは必須です。

1. 前提条件と PARMLIB 更新を確認および調整します。詳細については、88 ページの『前提条件』を参照してください。
2. Developer for System z 構成ファイルをカスタマイズします。詳細については、以下を参照してください。
 - 88 ページの『SCLMDT 用の ISPF.conf の更新』
 - 89 ページの『SCLMDT 用の rsed.envvars の更新』
3. オプションとして、ロング/ショート・ネーム変換サポートを定義します。詳細については、90 ページの『(オプション) ロング/ショート・ネーム変換』を参照してください。
4. オプションとして、JAVA/J2EE ビルド・サポートを使用するために Ant をインストールおよびカスタマイズします。詳細については、92 ページの『(オプション) Ant のインストールおよびカスタマイズ』を参照してください。
5. SCLM を更新して、SCLMDT 固有の部分を実行します。詳細については、93 ページの『SCLMDT 用の SCLM の更新』を参照してください。
6. オプションとして、SCLMDT 作業域の定期的な自動クリーンアップを設定します。詳細については、94 ページの『WORKAREA からの古いファイルの除去』を参照してください。

前提条件

必要な SCLM 保守のリストについては、357 ページの『付録 E. 必要条件』を参照してください。

この付録には、SCLM Developer Toolkit 内の JAVA/J2EE ビルドに必要な Ant 仕様も記載されています。

重要: SCLM Developer Toolkit には ISPF の TSO/ISPF クライアント・ゲートウェイを使用する必要があります。これは、z/OS 1.8 以上が必要であることを意味します。

19 ページの『PARMLIB の変更』の説明にあるように、SCLM Developer Toolkit を使用するには、システム設定の追加のカスタマイズが必要です。それらの変更内容は、以下のとおりです。

- (BPXPRMxx) z/OS UNIX ユーザー ID の 1 つ当たりの最大プロセス数を増やします。
- (PROGxx) SYS1.LINKLIB および REXX ランタイム REXX.V1R4M0.SEAGLPA または REXX.V1R4M0.SEAGALT の APF 許可を与えます。
- (PROGxx/LPALSTxx) ISP.SISPLPA、ISP.SISPLOAD、SYS1.LINKLIB、および REXX ランタイムを LINKLIST/LPALIB に入れます。

また、SCLM Developer Toolkit は SDSF または TSO **OUTPUT** コマンドを使用して、ジョブの完了状況とジョブの出力を取り出します。どちらの方法でも、以下の点にさらに注意が必要です。

- SDSF は、別途オーダーし、インストールし、構成する必要があります。また、JES2 を使用する必要もあります。
- TSO **OUTPUT** コマンドのデフォルトの設定では、ユーザーは自分のユーザー ID で始まるジョブ出力だけを取り出すことができます。**OUTPUT** 機能を完全に使用する場合は、ユーザーが自分の所有するジョブ出力をそれが自分のユーザー ID で始まっていなくても取り出すことができるように、サンプルの TSO/E 出口 IKJEFF53 を変更する必要がある場合があります。この出口について詳しくは、「TSO/E カスタマイズ」(SA88-8629)を参照してください。

ユーザーには、z/OS UNIX ディレクトリー /tmp/ および /var/rdz/WORKAREA/ に対する READ、WRITE、および EXECUTE 権限が必要です。ディレクトリー WORKAREA/ は /var/rdz/ に置かれます。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

SCLMDT 用の ISPF.conf の更新

SCLM Developer Toolkit は標準の ISPF/SCLM スケルトンを使用するため、スケルトン・ライブラリー ISP.SISPSLIB が ISPF.conf 内の ISPSLIB 連結に割り振られるようにしてください。ISP.SISPSENU データ・セットの使用はオプションです。

ISPF.conf は /etc/rdz/ に置かれます。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除き

ます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。このファイルは、TSO **OEDIT** コマンドで編集できます。

注: 変更は、更新後にホストに接続するすべてのクライアントについて有効になります。

次のサンプル・コードは `ISPF.conf` ファイルを示しています。このファイルは、使用するシステム環境に合わせてカスタマイズする必要があります。コメント行は、アスタリスク (*) で始まります。同じ行にある連結にデータ・セットを追加し、名前同士をコンマ (,) で分離します。`ISPF.conf` のカスタマイズの詳細については、51 ページの『`ISPF.conf`、ISPF の TSO/ISPF クライアント・ゲートウェイ構成ファイル』を参照してください。

```
* REQUIRED:
sysproc=ISP.SISPCLIB,FEK.SFEKPROC
ispmlib=ISP.SISPMENU
isptlib=ISP.SISPTENU
ispplib=ISP.SISPPENU
ispslib=ISP.SISPSLIB

* OPTIONAL:
*allocjob = FEK.#CUST.CNTL(CRAISPRX)
*ISPF_timeout = 900
```

図 21. `SCLMDT` 用の `ISPF.conf` の更新

注:

- TSO 環境をカスタマイズするために、独自の DD のようなステートメントとデータ・セット連結を追加し、TSO ログオン・プロシーチャーを模倣することができます。詳細については、283 ページの『第 16 章 TSO 環境のカスタマイズ』を参照してください。
- バッチ・ビルドを行う場合は、`ISPF/SCLM` スケルトン・ライブラリーの前に、カスタマイズされたバージョンの `FLMLIBS` スケルトンが連結されるようにしてください。

```
ispslib=hlq.USERSKEL,ISP.SISPSLIB
```

SCLMDT 用の `rsed.envvars` の更新

`SCLM Developer Toolkit` は、`rsed.envvars` の中で設定されたいくつかのディレクティブを使用して、データ・セットおよびディレクトリーを見つけます。

`rsed.envvars` は `/etc/rdz/` に置かれます。ただし、ジョブ `FEK.SFEKSAMP` (`FEKSETUP`) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。このファイルは、TSO **OEDIT** コマンドで編集できます。

注: 加えた変更があれば、それを取得するために `RSED` 開始タスクを再始動する必要があります。

次のコード・サンプルは、`rsed.envvars` ファイル内の `SCLMDT` ディレクティブを示しています。このファイルは、使用するシステム環境に合わせてカスタマイズする必要があります。`rsed.envvars` のカスタマイズの詳細については、35 ページの

『rsed.envvars、RSE 構成ファイル』を参照してください。

```
_SCLMDT_CONF_HOME=/var/rdz/sclmdt
#STEPLIB=$STEPLIB:FEK.SFEKAUTH:FEK.SFEKLOAD
#_SCLMDT_TRANTABLE=FEK.#CUST.LSTRANS.FILE
#ANT_HOME=/usr/lpp/Apache/Ant/apache-ant-1.7.1
_SCLMDT_BASE_HOME=$RSE_HOME
_SCLMDT_WORK_HOME=$_CMDSESV_WORK_HOME
CGI_DTWORK=$_SCLMDT_WORK_HOME
```

図 22. SCLMDT 用の rsed.envvars の更新

(オプション) ロング/ショート・ネーム変換

SCLM Developer Toolkit では、ロング・ネーム・ファイル (8 文字を超える名前または大/小文字混合の名前を持つファイル) を SCLM に保管することができます。これは、長いファイル名から SCLM で使用される 8 文字のメンバー名へのマッピングを含む VSAM ファイルを使用して達成されます。

注:

- z/OS 1.8 より前のバージョンの場合、この機能は APAR OA11426 に対応する基本 ISPF/SCLM PTF によって提供されます。
- ロング/ショート・ネーム変換は、IBM SCLM Administrator Toolkit などの、他の SCLM 関連製品でも使用されます。

LSTRANS.FILE (ロング/ショート・ネーム変換 VSAM) の作成

ISPF サンプル・ライブラリー ISP.SISPSAMP 内のサンプル・メンバー FLM02LST をカスタマイズして実行依頼し、ロング/ショート・ネーム変換 VSAM を作成します。本資料の構成ステップでは、以下のサンプルのセットアップ JCL に示すように、この VSAM に FEK.#CUST.LSTRANS.FILE という名前を付けるものとします。

```

//FLM02LST JOB <job parameters>
/*
/* CAUTION: This is neither a JCL procedure nor a complete job.
/* Before using this sample, you will have to make the following
/* modifications:
/* 1. Change the job parameters to meet your system requirements.
/* 2. Change ***** to the volume that will hold the VSAM.
/* 3. Change all references of FEK.#CUST.LSTRANS.FILE to
/*     match your naming convention for the SCLM translate VSAM.
/*
//CREATE EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
DELETE FEK.#CUST.LSTRANS.FILE
SET MAXCC=0
DEFINE CLUSTER(NAME(FEK.#CUST.LSTRANS.FILE) -
               VOLUMES(*****)) -
               RECORDSIZE(58 2048) -
               SHAREOPTIONS(3 3) -
               CYLINDERS(1 1) -
               KEYS(8 0) -
               INDEXED) -
DATA (NAME(FEK.#CUST.LSTRANS.FILE.DATA)) -
INDEX (NAME(FEK.#CUST.LSTRANS.FILE.INDEX))

/* DEFINE ALTERNATE INDEX WITH NONUNIQUE KEYS -> ESDS */

DEFINE ALTERNATEINDEX(-
               NAME(FEK.#CUST.LSTRANS.FILE.AIX) -
               RELATE(FEK.#CUST.LSTRANS.FILE) -
               RECORDSIZE(58 2048) -
               VOLUMES(*****)) -
               CYLINDERS(1 1) -
               KEYS(50 8) -
               UPGRADE -
               NONUNIQUEKEY) -
DATA (NAME(FEK.#CUST.LSTRANS.FILE.AIX.DATA)) -
INDEX (NAME(FEK.#CUST.LSTRANS.FILE.AIX.INDEX))

/*
/*
//PRIME EXEC PGM=IDCAMS,COND=(0,LT)
//SYSPRINT DD SYSOUT=*
//INITREC DD *
INITREC1
/*
//SYSIN DD *
REPRO INFILE(INITREC) -
      OUTDATASET(FEK.#CUST.LSTRANS.FILE)
IF LASTCC = 4 THEN SET MAXCC=0

BLDINDEX IDS(FEK.#CUST.LSTRANS.FILE) -
          ODS(FEK.#CUST.LSTRANS.FILE.AIX)

IF LASTCC = 0 THEN -
  DEFINE PATH (NAME(FEK.#CUST.LSTRANS.FILE.PATH) -
              PATHENTRY (FEK.#CUST.LSTRANS.FILE.AIX))
/*

```

図 23. FLM02LST - ロング/ショート・ネーム変換セットアップ JCL

注: 169 ページの『第 10 章 セキュリティーに関する考慮事項』の説明のように、ユーザーには、この VSAM データ・セットに対する UPDATE 権限が必要です。

ロング/ショート・ネーム変換用の rsed.envvars の更新

ロング/ショート・ネーム変換を使用する前に、rsed.envvars 環境変数 _SCLMDT_TRANTABLE をコメント解除し、ロング/ショート・ネーム変換 VSAM の名前に一致するように設定します。

rsed.envvars は /etc/rdz/ に置かれます。ただし、ジョブ FEK.SFEKSAMP (FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。このファイルは、TSO OEDIT コマンドで編集できます。

注: 加えた変更があれば、それを取得するために RSED 開始タスクを再始動する必要があります。

(オプション) Ant のインストールおよびカスタマイズ

このステップは、SCLM で JAVA/J2EE ビルド・サポートを使用する場合にのみ、必要となります。

Apache Ant はオープン・ソースの Java ビルド・ツールであり、<http://ant.apache.org/> からダウンロードできます。Ant はテキスト・ファイルとスクリプトからなり、それらは ASCII フォーマットで配布されています。このため、z/OS UNIX で実行するためには、ASCII/EBCDIC 変換が必要です。

以下のステップを実行して、Ant を z/OS に実装し、Developer for System z に対して定義します。

- 最新の Ant 圧縮ファイルをバイナリー・フォーマットで z/OS UNIX ファイル・システムにダウンロードします。.zip バージョンの ANT をダウンロードすることを推奨します。接尾部の形式が tar.gz または tar.bz2 のバージョンを抽出すると、z/OS で問題が起きる可能性があるからです。
- z/OS UNIX コマンド行セッションを開き、TSO OMVS コマンドを使用するなどして、インストールを続行します。
- **mkdir -p /home-dir** コマンドを使用して Ant インストールのホーム・ディレクトリを作成し、それを **cd /home-dir** コマンドで現行ディレクトリにします。
- JAR 抽出コマンド **jar -xf apache-ant-1.7.1.zip** を使用して、ファイルを現行ディレクトリに抽出してください。jar コマンドを使用するには、Java bin ディレクトリがローカルの z/OS UNIX パスに入っている必要があります。そうでない場合は、コマンドを Java bin ロケーションで完全修飾してください (例えば、/usr/lpp/java/J5.0/bin/jar -xf apache-ant-1.7.1.zip)。
- すべての Ant テキスト・ファイルを EBCDIC に変換するため、サンプル・スクリプトの /usr/lpp/rdz/samples/BWBTRANT を (オプションとしてカスタマイズし) 実行します。

注: このスクリプトは 1 回だけ実行してください。複数回実行すると、Ant インストールが壊れます。

- 変換が成功したかどうかを検査するために、ANT ディレクトリ内のテキスト・ファイル、例えば apache-ant-1.7.1/README などを見つけて参照します。ファイルが読み取り可能であれば、変換は正常に行われています。

- **chmod -R 755 *** コマンドを使用して、すべてのユーザーが ANT ディレクトリ内のファイルを READ および EXECUTE できるようにします。
- Ant を使用する前に、rsed.envvars の環境変数 JAVA_HOME および ANT_HOME を設定します。
 - JAVA_HOME は Java ホーム・ディレクトリーを指す必要があります。次に例を示します。
`JAVA_HOME=/usr/lpp/java/IBM/J5.0`
 - ANT_HOME は、次の例のように、Ant ホーム・ディレクトリーを指す必要があります。
`ANT_HOME=/usr/lpp/Apache/Ant/apache-ant-1.7.1`

次に例を示します。

- TSO OMVS
- `mkdir -p /usr/lpp/Apache/Ant`
- `cd /usr/lpp/Apache/Ant`
- `jar -xf /u/userid/apache-ant-1.7.1`
- `/usr/lpp/rdz/samples/BWBTRANT`
- `cat ./apache-ant-1.7.1/README`
- `chmod -R 755 *`
- `oedit /etc/rsed.envvars`

Ant の初期化が正常に行われたことをテストするには、次のようにします。

- Ant および Java bin ディレクトリーを環境変数 PATH に追加します。

例:

```
export PATH=/usr/lpp/Apache/Ant/apache-ant-1.7.1/bin:$PATH
export PATH=/usr/lpp/java/IBM/J5.0/bin:$PATH
```

- 正常にインストールが終了したら、**ant -version** を実行して、バージョンを表示します。

例:

```
ant -version
```

注: この方法による PATH ステートメントの設定はテスト時に必要なものであって、実際の運用に際しては不要です。

SCLMDT 用の SCLM の更新

SCLM 自体も、SCLM Developer Toolkit を処理するためにはカスタマイズが必要です。以下の必要なカスタマイズ・タスクの詳細については、「*IBM Rational Developer for System z SCLM Developer Toolkit 管理者ガイド*」(SC88-5664) を参照してください。

- JAVA/J2EE サポート用の言語変換プログラムを定義する
- JAVA/J2EE サポート用の SCLM タイプを定義する

カスタマイズ・タスクおよびプロジェクト定義タスクを完了するためには、SCLM 管理者は、表 13 に示す Developer for System z のカスタマイズ可能値を知っている必要があります。

表 13. SCLM 管理者チェックリスト

説明	<ul style="list-style-type: none"> デフォルト値 正解の入手先 	値
Developer for System z サンプル・ライブラリー	<ul style="list-style-type: none"> FEK.SFEKSAMV SMP/E インストール 	
Developer for System z サンプル・ディレクトリー	<ul style="list-style-type: none"> /usr/lpp/rdz/samples SMP/E インストール 	
Java bin ディレクトリー	<ul style="list-style-type: none"> /usr/lpp/java/J5.0/bin rsed.envvars - \$JAVA_HOME/bin 	
Ant bin ディレクトリー	<ul style="list-style-type: none"> /usr/lpp/Apache/Ant/apache-ant-1.7.1/bin rsed.envvars - \$ANT_HOME/bin 	
WORKAREA ホーム・ディレクトリー	<ul style="list-style-type: none"> /var/rdz rsed.envvars - \$_CMDSESV_CONF_HOME 	
SCLMDT プロジェクト構成ホーム・ディレクトリー	<ul style="list-style-type: none"> /var/rdz/sclmdt rsed.envvars - \$_SCLMDT_CONF_HOME 	
ロング/ショート・ネーム変換 VSAM	<ul style="list-style-type: none"> FEK.#CUST.LSTRANS.FILE rsed.envvars - \$_SCLMDT_TRANTABLE 	

WORKAREA からの古いファイルの除去

SCLM Developer Toolkit と ISPF の TSO/ISPF クライアント・ゲートウェイは同じ WORKAREA を共用し、この WORKAREA は定期的なクリーンアップを必要とする場合があります。それについての詳細は、115 ページの『(オプション) WORKAREA クリーンアップ』を参照してください。

第 6 章 (オプション) その他のカスタマイズ・タスク

このセクションは、さまざまなオプションのカスタマイズ・タスクを結合したものです。求めるサービスを構成するには、該当するセクションの説明に従ってください。

- 『(オプション) DB2 ストアード・プロシージャ』
- 97 ページの『(オプション) エンタープライズ・サービス・ツール (EST) サポート』
- 98 ページの『(オプション) CICS 双方向言語サポート』
- 99 ページの『(オプション) 診断用 IRZ エラー・メッセージ』
- 100 ページの『(オプション) RSE SSL 暗号化』
- 102 ページの『(オプション) RSE トレース』
- 104 ページの『(オプション) ホスト・ベース・プロパティ・グループ』
- 105 ページの『(オプション) ホスト・ベース・プロジェクト』
- 106 ページの『(オプション) File Manager Integration』
- 108 ページの『(オプション) 編集不可能文字』
- 109 ページの『(オプション) REXEC (または SSH) の使用』
- 111 ページの『(オプション) TSO コマンド・サービス用の APPC トランザクション』
- 115 ページの『(オプション) WORKAREA クリーンアップ』

(オプション) DB2 ストアード・プロシージャ

このカスタマイズ・タスクを完了するには、WLM 管理者および DB2 管理者の支援が必要になります。このタスクには、以下のリソースまたは特殊なカスタマイズ・タスクが必要です。

- WLM 更新
- 新規 PROCLIB メンバー
- DB2 更新

Developer for System z には、Developer for System z クライアントから COBOL および PL/I ストアード・プロシージャをビルドするための、サンプルの DB2 ストアード・プロシージャ (PL/I および COBOL ストアード・プロシージャ・ビルダー) が用意されています。

注: サンプルのメンバー ELAXM* は、FEK.#CUST.JCL および FEK.#CUST.PROCLIB の中に置かれます。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

ワークロード・マネージャー (WLM) の変更

ワークロード管理 (WLM) パネルを使用して、アプリケーション環境を、PL/I および COBOL ストアード・プロシージャ・ビルダー用の WLM アドレス・スペー

スの JCL プロシージャーに関連付けます。その方法については、「MVS 計画: ワークロード管理」(SA88-8574) を参照してください。

注: PL/I および COBOL ストアード・プロシージャー・ビルダー用に WLM で新しいアプリケーション環境を作成するか、既存の環境に必要な定義を追加することができます。

PROCLIB の変更

サンプルのストアード・プロシージャー・タスク FEK.#CUST.PROCLIB(ELAXMSAM) を、このメンバー内で説明されているようにカスタマイズし、SYS1.PROCLIB にコピーしてください。下記のコード・サンプルに示すように、以下のものを提供する必要があります。

- このストアード・プロシージャー用に WLM で定義されたアプリケーション環境の名前。
- DB2 サブシステム名
- 各種データ・セットの高位修飾子

```
//ELAXMSAM PROC RGN=0M,
//          NUMTCB=1,
//          APPLENV=#w1mwd4z,
//          DB2SSN=#ssn,
//          DB2PRFX='DSN810',
//          COBPRFX='IGY.V3R4M0',
//          PLIPRFX='IBMZ.V3R6M0',
//          LIBPRFX='CEE',
//          LODPRFX='FEK'
//*
//DSNX9WLM EXEC PGM=DSNX9WLM,REGION=&RGN,TIME=NOLIMIT,DYNAMNBR=10,
//          PARM='&DB2SSN,&NUMTCB,&APPLENV'
//STEPLIB DD DISP=SHR,DSN=&DB2PRFX..SDSNEXIT
//          DD DISP=SHR,DSN=&DB2PRFX..SDSNLOAD
//          DD DISP=SHR,DSN=&LIBPRFX..SCEERUN
//          DD DISP=SHR,DSN=&COBPRFX..SIGYCOMP
//          DD DISP=SHR,DSN=&PLIPRFX..SIBMZCMP
//SYSEXEC DD DISP=SHR,DSN=&LODPRFX..SFEKPROC
//SYSTSPRT DD SYSOUT=*
//CEEDUMP DD SYSOUT=*
//SYSABEND DD DUMMY
//SYSUT1 DD UNIT=SYSALLDA,SPACE=(CYL,(1,1))
//SYSUT2 DD UNIT=SYSALLDA,SPACE=(CYL,(1,1))
//SYSUT3 DD UNIT=SYSALLDA,SPACE=(CYL,(1,1))
//SYSUT4 DD UNIT=SYSALLDA,SPACE=(CYL,(1,1))
//SYSUT5 DD UNIT=SYSALLDA,SPACE=(CYL,(1,1))
//SYSUT6 DD UNIT=SYSALLDA,SPACE=(CYL,(1,1))
//SYSUT7 DD UNIT=SYSALLDA,SPACE=(CYL,(1,1))
//*
```

図 24. ELAXMSAM - DB2 ストアード・プロシージャー・タスク

注:

- DB2 ストアード・プロシージャーは、FEK.SFEKPROC にある REXX exec ELAXMREX を使用します。可能な SMP/E 保守を自動的にアクティブにしたい場合は、このロケーションを変更しないでください。
- メンバー ELAXMSAM および ELAXMREX を名前変更したい場合は、291 ページの『第 17 章 複数のインスタンスの実行』を参照してください。

DB2 の変更

データ・セット FEK.#CUST.JCL 内のサンプル・メンバー ELAXMJCL をカスタマイズおよび実行依頼し、DB2 に対してストアード・プロシーチャーを定義します。カスタマイズの手順については、メンバー内のドキュメンテーションを参照してください。

```
//ELAXMJCL JOB <job parameters>
//JOBPROC JCLLIB ORDER=(#hlq.SDSNPROC)
//JOBLIB DD DISP=SHR,DSN=#hlq.SDSNEXIT
// DD DISP=SHR,DSN=#hlq.SDSNLOAD
//*
//RUNTIAD EXEC PGM=IKJEFT01,DYNAMNBR=20
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
DSN S(#ssn) R(1) T(1)
RUN PROGRAM(DSNTIAD) PLAN(DSNTIAD) -
LIB('#hlq.RUNLIB.LOAD')
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
CREATE PROCEDURE SYSPROC.ELAXMREX
( IN FUNCTION_REQUEST VARCHAR(20) CCSID EBCDIC
, IN SQL_ROUTINE_NAME VARCHAR(27) CCSID EBCDIC
, IN SQL_ROUTINE_SOURCE VARCHAR(32672) CCSID EBCDIC
, IN BIND_OPTIONS VARCHAR(1024) CCSID EBCDIC
, IN COMPILE_OPTIONS VARCHAR(255) CCSID EBCDIC
, IN PRECOMPILE_OPTIONS VARCHAR(255) CCSID EBCDIC
, IN PRELINK_OPTIONS VARCHAR(32672) CCSID EBCDIC
, IN LINK_OPTIONS VARCHAR(255) CCSID EBCDIC
, IN ALTER_STATEMENT VARCHAR(32672) CCSID EBCDIC
, IN SOURCE_DATASETNAME VARCHAR(80) CCSID EBCDIC
, IN BUILDOWNER VARCHAR(8) CCSID EBCDIC
, IN BUILDUTILITY VARCHAR(18) CCSID EBCDIC
, OUT RETURN_VALUE VARCHAR(255) CCSID EBCDIC )
PARAMETER STYLE GENERAL RESULT SETS 1
LANGUAGE REXX EXTERNAL NAME ELAXMREX
COLLID DSNREXCS WLM ENVIRONMENT ELAXMSAM
PROGRAM TYPE MAIN MODIFIES SQL DATA
STAY RESIDENT NO COMMIT ON RETURN NO
ASUTIME NO LIMIT SECURITY USER;

COMMENT ON PROCEDURE SYSPROC.ELAXMREX IS
'PLI & COBOL PROCEDURE PROCESSOR (ELAXMREX), INTERFACE LEVEL 0.01';

GRANT EXECUTE ON PROCEDURE SYSPROC.ELAXMREX TO PUBLIC;
/*
```

図 25. ELAXMJCL - DB2 ストアード・プロシーチャー定義

注: CREATE PROCEDURE ステートメント内の WLM ENVIRONMENT 節で、PL/I および COBOL ストアード・プロシーチャー・ビルダー用に定義してある WLM 環境プロシーチャーの名前 (デフォルトでは ELAXMSAM) が指定されていることを確認してください。

(オプション) エンタープライズ・サービス・ツール (EST) サポート

このカスタマイズ・タスクには、支援や特殊リソース、または特殊なカスタマイズ・タスクは必要ありません。

Developer for System z クライアントには、エンタープライズ・サービス・ツール (EST) と呼ばれるコード生成コンポーネントがあります。生成されるコードは、そのタイプに応じて、Developer for System z ホスト・インストールが提供する機能に依存します。これらのホスト機能を使用可能にする方法を、以下のセクションで説明しています。

- 79 ページの『第 4 章 (オプション) Application Deployment Manager』
- 『(オプション) CICS 双方向言語サポート』
- 99 ページの『(オプション) 診断用 IRZ エラー・メッセージ』

注: エンタープライズ・サービス・ツール (EST) には、サービス・フロー・モデラー (SFM) や エンタープライズ用 XML サービス (XSE) などの複数のツールが含まれます。

(オプション) CICS 双方向言語サポート

このカスタマイズ・タスクを完了するには、CICS 管理者の支援が必要になります。このタスクには、以下のリソースまたは特殊なカスタマイズ・タスクが必要です。

- CICS 領域 JCL を更新する
- CICS に対してプログラムを定義する

Developer for System z エンタープライズ・サービス・ツール (EST) コンポーネントは、さまざまな形式のアラビア語およびヘブライ語のインターフェース・メッセージをサポートしているほか、すべてのエディターおよびビューで双方向言語データの表示と編集をサポートしています。端末アプリケーションでは、左から右と、右から左の両方の画面がサポートされ、数値フィールド、および画面とは反対の向きのフィールドもサポートされます。

そのほかの双方向言語フィーチャーおよび機能には、以下のものがあります。

- EST サービス・リクエストは、インターフェース・メッセージの双方向属性を動的に指定します。
- サービス・フロー内の双方向データ処理は、双方向属性 (テキスト・タイプ、テキスト方向、数値スワッピング、および対称スワッピング) に基づいています。これらの属性は、インターフェース・フローと端末フローのどちらの場合でも、さまざまなフロー作成のステージで指定できます。
- EST 生成ランタイム・コードには、異なる双方向属性を持つメッセージ内のフィールド間でのデータの変換が含まれます。

さらに、EST 生成コードは、CICS SFR (サービス・フロー・ランタイム) 以外の環境での bidi 変換をサポートできます。その一例がバッチ・アプリケーションです。EST 生成ウィザードで適切な bidi 変換オプションを指定し、生成されたプログラムを適切な双方向変換ライブラリー FEK.SFEKLOAD とリンクすることにより、EST 生成プログラムに、双方向変換ルーチンの呼び出しを組み込ませることが出来ます。

CICS 双方向言語サポートをアクティブにするには、以下のタスクを実行します。

1. FEK.SFEKLOAD ロード・モジュール FEJBDCMP および FEJBDTRX を CICS RPL 連結 (DD ステートメント DFHRPL) の中に置きます。この作業は、適用された保守が自動的に CICS で使用可能になるように、インストール・データ・セットを連結に追加する方法で行ってください。

注: インストール・データ・セットを連結せず、モジュールを新規または既存のデータ・セットにコピーする場合は、このモジュールが DLL であり、PDSE ライブラリー内に存在しなければならないことに留意してください。

2. 適切な CEDA コマンドを使用して、FEJBDCMP および FEJBDTRX をプログラムとして CICS に対して定義します。次に例を示します。

```
CEDA DEF PROG(FEJBDCMP) LANG(LE) G(xxx)
CEDA DEF PROG(FEJBDTRX) LANG(LE) G(xxx)
```

(オプション) 診断用 IRZ エラー・メッセージ

このカスタマイズ・タスクには、支援は必要ありませんが、以下のリソースまたは特殊なカスタマイズ・タスクが必要となります。

- LINKLIST 更新
 - CICS 領域 JCL を更新する
-

Developer for System z クライアントには、エンタープライズ・サービス・ツール (EST) と呼ばれるコード生成コンポーネントがあります。EST によって生成されたコードが診断用エラー・メッセージを発行するためには、FEK.SFEKLOAD ロード・ライブラリー内のすべての IRZ* および IIRZ* モジュールを、その生成コードが使用できるようにする必要があります。EST では、以下の環境向けにコードを生成できます。

- CICS
- IMS
- MVS バッチ

生成コードが CICS トランザクションで実行される場合は、FEK.SFEKLOAD 内のすべての IRZ* および IIRZ* モジュールを、CICS 領域の DFHRPL DD に追加します。この作業は、適用された保守が自動的に CICS で使用可能になるように、インストール・データ・セットを連結に追加する方法で行ってください。

それ以外の状態では、STEPLIB または LINKLIST を通じて、FEK.SFEKLOAD 内のすべての IRZ* および IIRZ* モジュールを使用可能にします。この作業は、適用された保守が自動的に CICS で使用可能になるように、インストール・データ・セットを連結に追加する方法で行ってください。

STEPLIB を使用する場合は、LINKLIST によって使用できないモジュールを、コードを実行するタスクの STEPLIB ディレクティブで定義する必要があります。

ロード・モジュールを使用できず、生成コードでエラーが発生した場合は、次のエラー・メッセージが発行されます。

```
IRZ9999S Failed to retrieve the text of a Language Environment runtime
message. Check that the Language Environment runtime message module for
facility IRZ is installed in DFHRPL or STEPLIB.
```

(オプション) RSE SSL 暗号化

このカスタマイズ・タスクを完了するには、セキュリティー管理者の支援が必要になります。このタスクには、以下のリソースまたは特殊なカスタマイズ・タスクが必要です。

- LINKLIST 更新
- プログラム制御データ・セットを追加するセキュリティー規則
- (オプション) SSL の証明書を追加するセキュリティー規則

外部 (クライアント/ホスト) 通信を SSL (Secure Sockets Layer) で暗号化できます。このフィーチャーは、デフォルトでは使用不可に設定され、`ssl.properties` 内の設定によって制御されます。

`ssl.properties` は `/etc/rdz/` に置かれます。ただし、ジョブ `FEK.SFEKSAMP(FEKSETUP)` をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。このファイルは、**TSO OEDIT** コマンドで編集できます。変更を有効にするには、RSE の再始動が必要であることに注意してください。

クライアントは接続のセットアップ時には RSE デーモンと通信し、実際のセッションのときは RSE サーバーと通信します。どちらのデータ・ストリームも、SSL を使用可能にした場合は暗号化されます。

RSE デーモンおよび RSE サーバーは、両者間のアーキテクチャーの違いから、証明書の保管に関して異なるメカニズムをサポートしています。これは、RSE デーモンと RSE サーバーの両方に SSL 定義が必要であることを意味しています。RSE デーモンと RSE サーバーが同じ証明書管理方式を使用する場合は、共用証明書を使用できます。

表 14. SSL 証明書の保管メカニズム

証明書ストレージ	作成者および管理者	RSE デーモン	RSE サーバー
鍵リング	SAF 準拠のセキュリティー製品	サポート	サポート
鍵データベース	z/OS UNIX の gskkyman	サポート	/
鍵ストア	Java の keytool	/	サポート

注:

- SAF 準拠の鍵リングは、証明書の管理に推奨される方式です。
- SAF 準拠の鍵リングでは、証明書の秘密鍵が、セキュリティー・データベースに保管されるか、または System z 暗号化ハードウェアとのインターフェースである ICSF を使用して保管されます。ICSF へのアクセスは、CSFSERV セキュリティー・クラス内のプロファイルによって保護されます。

RSE デーモンは、System SSL の機能を使用して SSL を管理します。これは `SYS1.SIEALNKE` が、ご使用のセキュリティー・ソフトウェアによってプログラム制御されることが必要で、LINKLIST または `rsed.envvars` 内の `STEPLIB` ディレクティブを介して RSE から使用可能でなければならないことを意味しています。

次のコード・サンプルは `ssl.properties` ファイルを示しています。このファイルは、使用するシステム環境に合わせてカスタマイズする必要があります。US コード・ページを使用する場合、コメント行はポンド記号 (#) で始まります。データ行には、ディレクティブとそれに割り当てられる値のみを入れることができ、その同じ行にコメントを入れることはできません。行の継続はサポートされていません。

```
# ssl.properties - SSL configuration file
enable_ssl=false

# Daemon Properties

#daemon_keydb_file=
#daemon_keydb_password=
#daemon_key_label=

# Server Properties

#server_keystore_file=
#server_keystore_password=
#server_keystore_label=
#server_keystore_type=JCECERAFKS
```

図 26. `ssl.properties` - SSL 構成ファイル

デーモンおよびサーバーのプロパティは、SSL を使用可能にする場合にのみ、設定する必要があります。SSL セットアップの詳細については、315 ページの『付録 A. SSL および X.509 認証のセットアップ』を参照してください。

enable_ssl

SSL 通信を使用可能または使用不可に設定します。デフォルトは `false` です。有効な設定値は、`true` と `false` だけです。

daemon_keydb_file

RACF (または、同様なセキュリティー製品) 鍵リング名。鍵リングを使用せず、**gskkyman** を使用して鍵データベースを作成した場合は、鍵データベース名を指定してください。SSL を使用可能にする場合は、このディレクティブをコメント解除してカスタマイズします。

daemon_keydb_password

鍵リングを使用する場合はコメント化またはブランクのまま残し、そうでない場合は鍵データベースのパスワードを指定します。SSL を使用可能にする場合、しかも **gskkyman** 鍵データベースを使用する場合は、このディレクティブをコメント解除してカスタマイズします。

daemon_key_label

鍵リングまたは鍵データベースで使用される証明書ラベル (デフォルトのものとして定義されていない場合)。デフォルトを使用する場合は、コメント化する必要があります。SSL を使用可能にする場合、しかもデフォルトのセキュリティー証明書を使用しない場合は、このディレクティブをコメント解除してカスタマイズします。

server_keystore_file

Java の **keytool** コマンドによって作成された鍵ストアの名前、または、`server_keystore_type=JCECERAFKS` の場合は RACF (または同様なセキュリティー製品) 鍵リング名。SSL を使用可能にする場合は、このディレクティブをコメント解除してカスタマイズします。

server_keystore_password

鍵リングを使用する場合はコメント化またはブランクのまま残し、そうでない場合は鍵ストアのパスワードを指定します。SSL を使用可能にする場合、しかも **keytool** 鍵ストアを使用する場合は、このディレクティブをコメント解除してカスタマイズします。

server_keystore_label

鍵リングまたは鍵ストアで使用される証明書ラベル。デフォルトは、最初に検出された有効な証明書です。SSL を使用可能にする場合、しかもデフォルトのセキュリティー証明書を使用しない場合は、このディレクティブをコメント解除してカスタマイズします。

server_keystore_type

鍵ストアのタイプ。デフォルトは JKS です。有効な値は、以下のとおりです。

表 15. 有効な鍵ストアのタイプ

キーワード	鍵ストアのタイプ
JKS	Java 鍵ストア
JCERACFKS	SAF 準拠の鍵リング。この場合は、証明書の秘密鍵がセキュリティー・データベースに保管されます。
JCECCARACFKS	SAF 準拠の鍵リング。この場合は、証明書の秘密鍵が System z 暗号化ハードウェアとのインターフェースである ICSF を使用して保管されます。

注: 本書を公開した時点では、IBM z/OS Java で JCECCARACFKS をサポートするためには、/usr/lpp/java/J5.0/lib/security/java.security ファイルを更新する必要があります。以下の行を追加する必要があります。

```
security.provider.1=com.ibm.crypto.hwdrCCA.provider.IBMJCECCA
```

更新後のファイルは以下のようになります。

```
security.provider.1=com.ibm.crypto.hwdrCCA.provider.IBMJCECCA
security.provider.2=com.ibm.jsse2.IBMJSSEProvider2
security.provider.3=com.ibm.crypto.provider.IBMJCE
security.provider.4=com.ibm.security.jgss.IBMJGSSProvider
security.provider.5=com.ibm.security.cert.IBMCertPath
security.provider.6=com.ibm.security.sasl.IBMSASL
```

(オプション) RSE トレース

このカスタマイズ・タスクには、支援や特殊リソース、または特殊なカスタマイズ・タスクは必要ありません。

Developer for System z は、問題解決の目的から、さまざまなレベルでの内部プログラム・フローのトレースをサポートしています。RSE、および RSE が呼び出すサービスの一部では、出力ログでの必要な詳細レベルを認識するために、rsecomm.properties 内の設定を使用します。

重要: これらの設定の変更は、パフォーマンス低下の原因になるため、IBM サポートの指示の下でのみ実行してください。

`rsecomm.properties` は `/etc/rdz/` に置かれます。ただし、ジョブ `FEK.SFEKSAMP(FEKSETUP)` をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。このファイルは、**TSO OEDIT** コマンドで編集できます。

次のコード・サンプルは、`rsecomm.properties` ファイルを示しています。このファイルは、トレースの必要性に合わせてカスタマイズできます。US コード・ページを使用する場合、コメント行はポンド記号 (#) で始まります。データ行には、ディレクティブとそれに割り当てられる値のみを入れることができ、その同じ行にコメントを入れることはできません。行の継続はサポートされていません。

```
# server.version - DO NOT MODIFY!
server.version=5.0.0

# Logging level
# 0 - Log error messages
# 1 - Log error and warning messages
# 2 - Log error, warning and info messages
debug_level=1

# Log location
# Log_To_StdOut
# Log_To_File
log_location=Log_To_File
```

図 27. `rsecomm.properties` - ロギング構成ファイル

server.version

ロギング・サーバー・バージョン。デフォルトは 5.0.0 です。変更しないでください。

debug_level

出力ログの詳細レベル。デフォルトは 1 です (エラー・メッセージおよび警告メッセージをログに記録します)。`debug_level` は複数のサービスの詳細レベルを (したがって複数の出力ファイルを) 制御することに注意してください。詳細レベルの増大は、パフォーマンス低下の原因になるため、IBM サポートの指示の下でのみ実行してください。このディレクティブによって制御されるログの詳細については、155 ページの『RSE トレース』を参照してください。

有効な値は以下のとおりです。

0	エラー・メッセージのみをログに記録します。
1	エラー・メッセージと警告メッセージをログに記録します。
2	エラー・メッセージ、警告メッセージ、および情報メッセージをログに記録します。

注: 133 ページの『第 8 章 オペレーター・コマンド』の説明にあるように、`debug_level` は **modify rsecommlog** オペレーター・コマンドで動的に変更できます。

log_location

RSE に関連したログの出力メディア。デフォルトは `Log_To_File` です。
RSE デーモン接続方式 (デフォルト) を使用する場合は、IBM サポートに指示された場合を除き、変更しないでください。

有効な値は以下のとおりです。

Log_To_File	ログ・メッセージをログ出力ディレクトリー内の別のファイルに送信します。 <ul style="list-style-type: none">• RSE デーモン: <code>daemonlog</code> 内の <code>rsedaemon.log</code>• RSE スレッド・プール: <code>daemonlog</code> 内の <code>rserver.log</code>• ユーザー: <code>userlog/.eclipse/RSE/\$LOGNAME</code> 内の <code>rsecomm.log</code>
Log_To_StdOut	ログ・メッセージを <code>stdout</code> に送信します。 <ul style="list-style-type: none">• RSE デーモン: <code>RSED</code> 開始タスク内の <code>DD STDOUT</code> へ転送される• RSE スレッド・プール: 未定義• ユーザー: <code>userlog/.eclipse/RSE/\$LOGNAME</code> 内の <code>stdout.log</code> へ転送される

`daemonlog` は `rsed.envvars` 内の `daemon.log` ディレクティブの値です。`daemon.log` ディレクティブがコメント化されているか存在しない場合は、`RSED` 開始タスクに割り当てられているユーザー ID のホーム・パスが使用されます。ホーム・パスはユーザー ID の OMVS セキュリティー・セグメントで定義されます。

ユーザー固有のログは、`userlog/.eclipse/RSE/$LOGNAME` に書き込まれます。ここで、`userlog` は `rsed.envvars` 内の `user.log` ディレクティブの値、`$LOGNAME` はログオン・ユーザー ID (大文字) です。`user.log` ディレクティブがコメント化されているか存在しない場合は、ユーザーのホーム・パスが使用されます。ホーム・パスはユーザー ID の OMVS セキュリティー・セグメントで定義されます。

(オプション) ホスト・ベース・プロパティー・グループ

このカスタマイズ・タスクには、支援や特殊リソース、または特殊なカスタマイズ・タスクは必要ありません。

Developer for System z クライアントは、各種プロパティー (例えば、COBOL ソース・コードをコンパイルするときに使用する COBOL コンパイラー・オプション) のデフォルト値を保持するプロパティー・グループを定義できます。Developer for System z には、いくつかのデフォルト値が組み込まれていますが、システム固有のカスタムのデフォルト値を定義することもできます。

カスタムのプロパティー・グループ構成ファイルおよびデフォルト値構成ファイルのロケーションは `/etc/rdz/` にある `propertiescfg.properties` の中で定義されています。ただし、ジョブ `FEK.SFEKSAMP(FEKSETUP)` をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの

ージの『カスタマイズのセットアップ』を参照してください。このファイルは、**TSO OEDIT** コマンドで編集できます。変更を有効にするには、RSE の再始動が必要であることに注意してください。

次のコード・サンプルは `propertiescfg.properties` ファイルを示しています。このファイルは、使用するシステム環境に合わせてカスタマイズする必要があります。US コード・ページを使用する場合、コメント行はポンド記号 (#) で始まります。データ行には、ディレクティブとそれに割り当てられる値のみを入れることができます。同じ行にコメントを入れることはできません。行の継続はサポートされていません。

```
#
# host based property groups - root configuration file
#
ENABLED=FALSE
RDZ-VERSION=7.5.0.0
PROPERTY-GROUP=/var/rdz/properties
DEFAULT-VALUES=/var/rdz/properties
```

図 28. `propertiescfg.properties` - ホスト・ベース・プロパティー・グループ構成ファイル

ENABLED

Developer for System z でプロパティー・グループ構成ファイルおよびデフォルト値構成ファイルを使用するかどうかを示します。デフォルトは `FALSE` です。有効な設定値は `TRUE` と `FALSE` だけです。

RDZ-VERSION

ホスト・ベース・プロパティー・グループを使用するための最小の Developer for System z クライアント・レベル。デフォルトは `7.5.0.0` です。変更しないでください。

PROPERTY-GROUP

プロパティー・グループ構成ファイルのロケーション。デフォルトは `/var/rdz/properties` です。

DEFAULT-VALUES

デフォルト値構成ファイルのロケーション。デフォルトは `/var/rdz/properties` です。

プロパティー・グループ構成ファイル (`propertygroups.xml`) およびデフォルト値構成ファイル (`defaultvalues.xml`) の詳しい作成方法については、Developer for System z インフォメーション・センター (<http://publib.boulder.ibm.com/infocenter/ratdevz/v7r6/index.jsp>) を参照してください。

(オプション) ホスト・ベース・プロジェクト

このカスタマイズ・タスクには、支援や特殊リソース、または特殊なカスタマイズ・タスクは必要ありません。

z/OS プロジェクトは、クライアント上の「z/OS プロジェクト」パースペクティブを通じて個別に定義するか、ホスト上で集中的に定義してクライアントヘユーザー単位で伝搬することができます。それらの「ホスト・ベースのプロジェクト」は、クライアント上で定義されたプロジェクトと外観も機能もまったく同じですが、ク

クライアントは、それらの構造、メンバー、およびプロパティを変更できず、ホストに接続している場合にのみ、それらのプロジェクトにアクセスできます。

プロジェクト定義のロケーションは `/etc/rdz/` にある `projectcfg.properties` の中で定義されています。ただし、ジョブ `FEK.SFEKSAMP(FEKSETUP)` をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。このファイルは、**TSO OEDIT** コマンドで編集できます。変更を有効にするには、**RSE** の再始動が必要であることに注意してください。

次のコード・サンプルは `projectcfg.properties` ファイルを示しています。このファイルは、使用するシステム環境に合わせてカスタマイズする必要があります。US コード・ページを使用する場合、コメント行はポンド記号 (#) で始まります。データ行には、ディレクティブとそれに割り当てられる値のみを入れることができます。同じ行にコメントを入れることはできません。行の継続はサポートされていません。

```
#
# host based projects - root configuration file
#
# WSED-VERSION - do not modify !
WSED-VERSION=7.0.0.0
# specify the location of the host based project definition files
PROJECT-HOME=/var/rdz/projects
```

図 29. `projectcfg.properties` - ホスト・ベース・プロジェクト構成ファイル

WSED-VERSION

ホスト・ベース・プロジェクトを使用するための最小の Developer for System z クライアント・レベル。デフォルトは 7.0.0.0 です。変更しないでください。

PROJECT-HOME

プロジェクト定義の基本ディレクトリー。デフォルトは `/var/rdz/projects` です。

注: ホスト・ベース・プロジェクトをアクティブにするには、`project.instance` ファイルが `/var/rdz/projects/USERID` 内にあることが必要です。ここで、`/var/rdz/projects` はプロジェクト定義ファイルのロケーションで、`USERID` は開発者がログオンに使用するユーザー ID (大文字) です。

ホスト・ベースのプロジェクトの詳細については、Developer for System z インフォメーション・センター (<http://publib.boulder.ibm.com/infocenter/ratdevz/v7r6/index.jsp>) を参照してください。

(オプション) File Manager Integration

このカスタマイズ・タスクを完了するには、セキュリティ管理者の支援が必要になります。このタスクには、以下のリソースまたは特殊なカスタマイズ・タスクが必要です。

- プログラム制御データ・セットを追加するセキュリティ規則
-

Developer for System z は、クライアントから IBM File Manager for z/OS 機能の限定セットへの直接アクセスをサポートしています。IBM File Manager for z/OS は、MVS データ・セット、z/OS UNIX ファイル、DB2、IMS および CICS データを処理するための包括的なツールを提供します。これらのツールは、ISPF でなじみ深いブラウザ、編集、コピー、および印刷ユーティリティを、アプリケーション開発者の必要性を満たすように機能拡張したものです。現行バージョンの Developer for System z では、MVS データ・セット (全タイプの VSAM を含みます) のブラウザと編集、MVS データ・セット・テンプレート (動的なテンプレートを含みます) の作成と編集、および拡張コピー・ユーティリティのみがサポートされています。

IBM File Manager for z/OS 製品は、別個にオーダーし、インストールし、構成する必要があることに注意してください。使用しているバージョンの Developer for System z に必要な File Manager のレベルについては、「*Rational Developer for System z 前提条件*」(SC88-4704) を参照してください。この製品のインストールとカスタマイズについては、本書には記載されていません。

Developer for System z および File Manager では、File Manager のサービスにアクセスするためのバッチ・インターフェースがサポートされなくなりました。現在は、File Manager リスナーを使用する必要があります。

注: File Manager 資料で説明されている通常のリスナー・セットアップ・タスクのほかに、Developer for System z ではサーバーの STEPLIB データ・セットをプログラムで制御する必要があります。

Developer for System z に必要な File Manager Integration 定義は、/etc/rdz/ にある FMEXT.properties に保管されています。ただし、ジョブ FEK.SFEKSAMP (FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。このファイルは、TSO **OEDIT** コマンドで編集できます。変更を有効にするには、RSE の再始動が必要であることに注意してください。

次のコード・サンプルは FMEXT.properties ファイルを示しています。このファイルは、使用するシステム環境に合わせてカスタマイズする必要があります。US コード・ページを使用する場合、コメント行はポンド記号 (#) で始まります。データ行には、ディレクティブとそれに割り当てられる値のみを入れることができます。同じ行にコメントを入れることはできません。行の継続はサポートされていません。

```
# File Manager Integration (FMI) Extension properties
#
enabled=false
fmlistenport=1960
```

図 30. FMEXT.properties - File Manager 構成ファイル

enabled

File Manager リスナーが同じホスト・システム上で使用可能かどうかを指定します。デフォルト値は false です。指定できる値は、true と false だけです。

fmlistenport

File Manager リスナーによって使用されるポート。デフォルトは 1960 です。このポート上の通信は、ご使用のホスト・マシンだけに限定されます。

(オプション) 編集不可能文字

このカスタマイズ・タスクには、支援や特殊リソース、または特殊なカスタマイズ・タスクは必要ありません。

一部の文字は、ホスト・コード・ページ (EBCDIC ベース) とクライアント・コード・ページ (ASCII ベース) の間で、正しく変換できません。Developer for System z クライアント・エディターは、`uchars.settings` ファイル内の定義を使用して、それらの編集不可能文字を識別します。エディターは、`uchars.settings` で識別された文字を含むデータ・セットが保存されるときに壊れることを防ぐために、そのデータ・セットをオープンする際に強制的に読み取り専用モードになります。

`uchars.settings` は `/etc/rdz/` に置かれます。ただし、ジョブ FEK.SFEKSAMP (FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。このファイルは、TSO **OEDIT** コマンドで編集できます。変更を有効にするには、RSE の再始動が必要であることに注意してください。また、IBM サポートから指示された場合以外、このファイルを変更しないことが推奨される点にも注意してください。

```
# uchars.settings - uneditable code points
#
*          *          0D 15 25;

# DBCS (Japanese, Korean & Chinese)
IBM-930   *          0D 15 1E 1F 25;
IBM-933   *          0D 15 1E 1F 25;
IBM-935   *          0D 15 1E 1F 25;
IBM-937   *          0D 15 1E 1F 25;
IBM-939   *          0D 15 1E 1F 25;
IBM-1390  *          0D 15 1E 1F 25;
IBM-1399  *          0D 15 1E 1F 25;
IBM-1364  *          0D 15 1E 1F 25;
IBM-1371  *          0D 15 1E 1F 25;
IBM-1388  *          0D 15 1E 1F 25;

# UNICODE
UTF-8     *          0D 0A;
UTF-16BE  *          0D 0A;
UTF-16LE  *          0D 0A;
UTF-16    *          0D 0A;
```

図 31. `uchars.settings` - 編集不可能文字構成ファイル

このファイルは、以下のフォーマットの複数の項目から構成されています。

HOST-CODEPAGE LOCAL-CODEPAGE HEX-CODEPOINTS ;

ここで、HEX-CODEPOINTS は、編集不可能文字を識別する 2 桁の 16 進コード・ポイントをブランクで区切ったリストです。このリストは、セミコロン (;) で終了する必要があります。

以下の構文規則が適用されます。

- US コード・ページを使用する場合、コメント行はポンド記号 (#) で始まります。
- データ行にはデータのみを入れることができ、その同じ行にコメントを入れることはできません。
- アスタリスク (*) を HOST-CODEPAGE か LOCAL-CODEPAGE、またはその両方に使用できます。これはワイルドカードとして機能し、すべてのコード・ページを表します。
- 具体的な項目は、汎用 (ワイルドカード) の項目より優先します。
- 「host-cp *」と「* local-cp」が両方とも指定され、「host-cp local-cp」によってオーバーライドされない場合、「host-cp *」が優先します。
- 同じコード・ページのペアが複数回指定された場合は、最後の項目が使用されます。

(オプション) REXEC (または SSH) の使用

このカスタマイズ・タスクには、支援や特殊リソース、または特殊なカスタマイズ・タスクは必要ありません。

REXEC (リモート実行) は、クライアントがホスト上でコマンドを実行できるようにする TCP/IP サービスです。SSH (セキュア・シェル) も同様なサービスですが、このサービスでは、すべての通信が SSL (Secure Sockets Layer) によって暗号化されます。Developer for System z は、いずれかのサービスを使用して、z/OS UNIX サブプロジェクトでリモート (ホスト・ベース) アクションを実行します。

また、REXEC (または SSH) を使用してホスト上で RSE サーバーを始動するように Developer for System z を構成することもできます。ただし、この方法で開始した接続では、接続ごとに別々の RSE サーバーが始動され、それぞれがかなりの量のシステム・リソースを使用することに注意してください。したがって、この代替の接続方式は、接続の数が少ない場合にしか実行できません。

また、REXEC (または SSH) 代替接続方式は RSE デーモンを迂回するため、本書で述べるすべてのホスト・サービスに必ずしもアクセスできるわけではありません。例えば、単一サーバー処理や監査にはアクセスできません。特定のホスト・サービスが REXEC 代替接続方式でサポートされているかどうかについては、IBM サポートにお問い合わせください

注: Developer for System z は、TSO バージョンでなく、z/OS UNIX バージョンの REXEC を使用します。

z/OS UNIX サブプロジェクト用のリモート (ホスト・ベース) アクション

z/OS UNIX サブプロジェクトのリモート (ホスト・ベースの) アクションを実行するには、ホスト上で REXEC または SSH がアクティブであることが必要です。REXEC/SSH がデフォルト・ポートを使用するように構成されていない場合、Developer for System z クライアントは z/OS UNIX サブプロジェクトに使用する正しいポートを定義する必要があります。これを行うには、「ウィンドウ」>「設定

...」>「z/OS ソリューション」>「USS サブプロジェクト」>「リモート・アクション・オプション」設定ページを選択します。どのポートが使用されるかについては、『REXEC (または SSH) のセットアップ』を参照してください。

代替 RSE 接続方式

Developer for System z クライアントは、REXEC (または SSH) を通じて RSE 接続を開始するために次の 2 つの値を認識している必要があります。

- `server.zseries` 始動スクリプトが置かれるディレクトリー。

`server.zseries` は `/etc/rdz/` に置かれます。ただし、ジョブ `FEK.SFEKSAMP(FEKSETUP)` をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

注: REXEC (または SSH) が代替ログオン方式として使用されることを防ぐために、`server.zseries` は `/etc/rdz` に自動的にコピーされないようになりました。この機能を使用する場合は、以下のサンプル・コマンドに示すように、このスクリプトを手動で `/usr/lpp/rdz/bin` にコピーする必要があります。

```
cp /usr/lpp/rdz/bin/server.zseries /etc/rdz
```

- 使用されているポート。

どのポートが使用されるかについては、『REXEC (または SSH) のセットアップ』を参照してください。

REXEC (または SSH) のセットアップ

REXEC (または SSH) をセットアップするために必要なステップについては、「*Communications Server IP 構成ガイド*」(SC88-8926) に説明があります。Developer for System z に固有なセットアップに関する考慮事項については、339 ページの『付録 C. INETD のセットアップ』を参照してください (Developer for System z に固有なセットアップ・ステップはありません)。

REXEC で使用される共通のポートは 512 です。これを確認するために、`/etc/inetd.conf` および `/etc/services` を調べて、使用されているポート番号を知ることができます。

- `/etc/inetd.conf` で、`rexecd` サーバー (7 番目のワード) のサービス名 (最初のワード。この例では `exec`) を見つけます。

```
exec stream tcp nowait OMVSKERN /usr/sbin/orexecd rexecd -LV
```

- `/etc/services` で、そのサービス名 (最初のワード) に接続しているポート (2 番目のワード。この例では 512) を見つけます。

```
exec      512/tcp      #REXEC      Command Server
```

同じ原則が SSH にも適用されます。その共通のポートは 22 で、サーバー名は `sshd` です。

注: `/etc/inetd.conf` と `/etc/services` は異なる名前にできます。詳細については、339 ページの『付録 C. INETD のセットアップ』を参照してください。

(オプション) TSO コマンド・サービス用の APPC トランザクション

このカスタマイズ・タスクを完了するには、APPC 管理者および WLM 管理者の支援が必要になります。このタスクには、以下のリソースまたは特殊なカスタマイズ・タスクが必要です。

- LINKLIST 更新
 - APPC トランザクション
 - WLM 更新
-

TSO コマンド・サービスは、APPC トランザクション・プログラム、FEKFRSRV として実装できます。このトランザクションはホスト・サーバーとして機能し、ワークステーションから発行された TSO コマンドおよび ISPF コマンドを実行します。クライアントは RSE を通じて FEKFRSRV と通信するので、ワークステーション上に APPC は必要ありません。各クライアントは、同時に複数のホストへのアクティブな接続を持つことができます。

注:

- デフォルトでは、Developer for System z は ISPF の TSO/ISPF クライアント・ゲートウェイを使用して TSO コマンド・サービスにアクセスします。
- APPC に詳しくない場合は、このセクションを続行する前に、349 ページの『付録 D. APPC のセットアップ』を参照してください。
- RSE は、TCP/IP REXX ソケット API を使用して FEKFRSRV と通信します。したがって、TCP/IP ロード・ライブラリーであるデフォルトの TCPIP.SEZALOAD を、rsed.envvars 内の LINKLIST または STEPLIB ディレクティブを通じて使用可能にする必要があります。
- 記述された変更を有効にするには、RSE を再始動する必要があります。

準備

- 以下の作業は前提条件であり、TSO コマンド・サーバーを構成する前に完了しておく必要があります。これらの作業については、示されている資料に説明があります。
 1. z/OS システム上で VTAM® をインストールし、構成し、始動します。詳細については、「*Communications Server IP SNA ネットワーク・インプリメンテーション・ガイド*」(SC88-8928) を参照してください。
 2. z/OS システム上で TCP/IP をインストールし、構成し、始動します。詳細については、329 ページの『付録 B. TCP/IP のセットアップ』を参照してください。
 3. APPC および APPC トランザクション・スケジューラー (ASCH) サブシステムを構成し、始動します。詳細については、349 ページの『付録 D. APPC のセットアップ』を参照してください。
- 次のサンプルの REXX を使用して、ISPF パネルを通じて APPC を管理することができます。

```

/* REXX -- APPC administration using ISPF panels */
address ISPEXEC
"LIBDEF ISPLIB DATASET ID('ICQ.ICQMLIB') STACK"
"LIBDEF ISPLIB DATASET ID('ICQ.ICQPLIB') STACK"
"LIBDEF ISPLIB DATASET ID('ICQ.ICQSLIB') STACK"
"LIBDEF ISPTLIB DATASET ID('ICQ.ICQTLIB') STACK"
address TSO "ALTLIB ACT APPLICATION(CLIST)",
            "DSN('ICQ.ICQCLIB') UNCOND QUIET"
"SELECT CMD(%ICQASRM0) NEWAPPL(ICQ) PASSLIB"
address TSO "ALTLIB DEACT APPLICATION(CLIST) QUIET"
"LIBDEF ISPLIB"
"LIBDEF ISPLIB"
"LIBDEF ISPLIB"
"LIBDEF ISPTLIB"
exit

```

図 32. APPC ISPF パネル用の REXX

注: このツールを使用して APPC トランザクションを非アクティブにできることに注意してください。その場合、トランザクションは、まだ存在していますが、接続を受け入れなくなります。

- APPC トランザクションの定義には、MVS オペレーティング・システムのさまざまな分野でのスキルが必要です。先へ進む前に、以下のチェックリストを使用して、経験のある管理者に相談してください。

表 16. APPC トランザクション・チェックリスト

専門知識	必須情報: <ul style="list-style-type: none"> • デフォルト値 • 正解の入手先 	値
APPC	TPDATA のデータ・セット名 <ul style="list-style-type: none"> • デフォルト: SYS1.APPCTP • 値は、SYS1.PARMLIB(APPCPMxx) にリストされています。 	
APPC	使用されるトランザクション名 (存在していない可能性がある) <ul style="list-style-type: none"> • デフォルト: FEKFRSRV • 既存のトランザクションは、APPC ISPF メニューで「TP プロファイル管理 (TP Profile Administration)」を選択すると照会できます。 	
APPC	使用される APPC トランザクション・クラス <ul style="list-style-type: none"> • デフォルト: A • APPC クラスは SYS1.PARMLIB(ASCHPMxx) 内で定義されています。 	
WLM/SRM	TSO パフォーマンス・グループおよびドメイン <ul style="list-style-type: none"> • IBM のデフォルトはありません (サイトに依存)。 	
RACF	すべての Developer for System z ユーザーは OMVS セグメントにアクセスできます (これは必須です)。 <ul style="list-style-type: none"> • IBM のデフォルトはありません (サイトに依存)。 • TSO RACF コマンド LU userid OMVS は、既存の個人用 OMVS セグメントを表示します。 	

表 16. APPC トランザクション・チェックリスト (続き)

専門知識	必須情報: • デフォルト値 • 正解の入手先	値
RACF	<p>すべての Developer for System z ユーザーは、hlq.SFEKPROC(FEKFRSRV) に対する READ アクセス権を持っている必要があります。</p> <ul style="list-style-type: none"> • IBM のデフォルトはありません (サイトに依存)。 • TSO RACF コマンド LD AUTHUSER DATASET('hlq.SFEKPROC.**') は、データ・セット・プロファイルに含まれるデータ・セットのユーザー、グループ、およびそれらのアクセス・レベルを表示します。 	

WLM/SRM 管理の詳細については、「MVS 計画: ワークロード管理」(SA88-8574)を参照してください。OMVS セグメントおよびデータ・セット保護プロファイルの詳細については、「Security Server RACF セキュリティー管理者のガイド」(SA88-8613)を参照してください。

実装

注: サンプル・メンバー FEKAPPC* は、FEK.#CUST.JCL に置かれます。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

1. 既存のトランザクション・クラスを使用していない場合は、APPC トランザクション・スケジューラーのスケジューリング情報 (クラス) を定義します。トランザクション・プログラム FEKFRSRV によって使用されるクラスの定義を、SYS1.PARMLIB(ASCHPMxx) に組み込んでください。このクラスは、サンプル JCL FEK.#CUST.JCL(FEKAPPCC) の中で使用されています。したがって、FEKAPPCC 内のクラスは、SYS1.PARMLIB(ASCHPMxx) の中で定義されているクラスに一致する必要があります。次に例を示します。

```
CLASSADD
  CLASSNAME(A)
  MAX(20)
  MIN(1)
  MSGLIMIT(200)
```

注:

- TSO コマンド・サービスを使用するには、SYS1.PARMLIB(ASCHPMxx) の OPTIONS セクションと TPDEFAULT セクションでデフォルト値を指定する必要があります。詳細については、349 ページの『付録 D. APPC のセットアップ』を参照してください。
- 使用される APPC トランザクション・クラスには、Developer for System z の各同時ユーザーに 1 つずつイニシエーターを許可できるだけの十分な数の APPC イニシエーターがあることが必要です。

2. コマンド・サーバーとして機能する APPC トランザクションを定義します。このトランザクションを定義する場合、サンプル JCL FEK.#CUST.JCL(FEKAPPC)を使用できます。この JCL をカスタマイズする方法については、JCL の中に説明があります。

注:

- a. このステップでトランザクション・プログラム名 (デフォルトは FEKFRSRV) を変更した場合は、35 ページの『rsed.envvars、RSE 構成ファイル』で説明されているように、新しい名前を rsed.envvars 内の _FEKFSCMD_TP_NAME_ に割り当てる必要があります。
 - b. APPC トランザクションは、FEK.SFEKPROC にある REXX exec FEKFRSRV を使用します。可能な SMP/E 保守を自動的にアクティブにしたい場合は、このロケーションを変更しないでください。
 - c. サンプル JCL は、トランザクションの表示用 (FEK.#CUST.JCL (FEKAPPC))、または削除用 (FEK.#CUST.JCL (FEKAPPCX)) にも提供されています。
3. RSE が APPC を使用できるようにするために、35 ページの『rsed.envvars、RSE 構成ファイル』の説明のように、rsed.envvars 内の RSE_JAVAOPTS="\$_RSE_JAVAOPTS -DTSO_SERVER=APPC" ディレクティブをコメント解除します。
 4. FEKFRSRV をワークロード・マネージャー (WLM) でドメインおよびパフォーマンス・グループに関連付けて、トランザクション・プログラムのディスパッチング優先順位を制御します。FEKFRSRV は、TSO コマンドを発行するので、TSO パフォーマンス・グループに割り当てる必要があります。
 5. システムにデフォルトの OMVS セグメントを定義するか、Developer for System z のユーザーごとに個別の OMVS セグメントを定義します。
 6. Developer for System z ユーザーに、Developer for System z TSO 実行可能ライブラリー FEK.SFEKPROC への READ アクセス権を与えます。

APPC の使用に関する考慮事項

- TSO コマンド・サービスに APPC を使用する場合、Developer for System z は TCP/IP が初期化時に正しいホスト名を持っているかどうか依存します。つまり、各種の TCP/IP 構成ファイルやリゾルバー構成ファイルが正しくセットアップされていなければなりません。TCP/IP およびリゾルバーのカスタマイズ情報については、329 ページの『付録 B. TCP/IP のセットアップ』、および『Communications Server IP 構成解説書』(SC88-8927) の『TCPIP.DATA 構成ステートメント』を参照してください。

IVP=IVP パラメーターを指定するか、fekfivpt インストール検査プログラム (IVP) と一緒に RSE デーモンを始動することにより、TCP/IP 構成をテストすることができます (117 ページの『第 7 章 インストール検査』を参照)。

- TSO コマンド・サービスに APPC を使用する場合、Developer for System z はオープンされた MVS ファイルごとに、ホスト限定通信用の追加ソケット (TCP/IP ポート) を必要とします。使用可能な任意のポートが使用されます。このポート選択メカニズムを変更することはできません。

- TSO コマンド・サービスに APPC を使用する場合に、MVS データ・セットの読み取りと書き込みを行うには、ソケット物理ファイル・システム・ドメインを使用する必要があります。このファイル・システムが正しく定義されていないか、十分な数のソケットを持っていない場合、ロック・マネージャー (FFS) が読み取り/書き込み要求を処理できないことがあります。166 ページの『MVS データ・セットのオープンの失敗』を参照してください。
- TSO コマンド・サービスに APPC を使用して ISPF の TSO/ISPF クライアント・ゲートウェイのセットアップを回避する場合は、SCLM Developer Toolkit などの他のサービスが TSO/ISPF クライアント・ゲートウェイに依存することに注意してください。
- 一般的な APPC の使用に関する考慮事項については、349 ページの『付録 D. APPC のセットアップ』を参照してください。

(オプション) WORKAREA クリーンアップ

このカスタマイズ・タスクには、支援や特殊リソース、または特殊なカスタマイズ・タスクは必要ありません。

ISPF の TSO/ISPF クライアント・ゲートウェイおよび SCLM Developer Toolkit 機能は WORKAREA ディレクトリーに一時作業ファイルを保管しますが、それらのファイルは、セッションが閉じる前に除去されます。ところが、処理中に通信エラーが発生した場合など、一時出力が残される場合があります。このため、時々 WORKAREA ディレクトリーを整理することをお勧めします。

z/OS UNIX には、ファイルが入っているディレクトリーとファイルの経過日数に基づいてファイルを削除する、`skulker` というシェル・スクリプトがあります。指定された日時にコマンドを実行する z/OS UNIX `cron` デーモンと結合すれば、定期的に WORKAREA ディレクトリーを空にする自動化ツールをセットアップできます。`skulker` スクリプトおよび `cron` デーモンの詳細については、「*UNIX System Services コマンド解説書*」(SA88-8641) を参照してください。

注: WORKAREA は `/var/rdz/` に置かれます。ただし、ジョブ FEK.SFEKSAMP (FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

第 7 章 インストール検査

製品のカスタマイズの完了後、この章で説明するインストール検査プログラム (IVP) を使用して、主要な製品コンポーネントのセットアップが正常であることを検査できます。

開始タスクの検査

JMON、JES ジョブ・モニター

JMON 開始タスク (またはユーザー・ジョブ) を開始します。DD STDOUT での開始情報は、次のメッセージで終わります。

```
JM200I Server initialization complete.
```

ジョブが戻りコード 66 で終了する場合は、FEK.SFEKAUTH に APF 許可がありません。

注: 他の IVP テストを続行する前に、JES ジョブ・モニターを始動してください。

LOCKD、ロック・デーモン

LOCKD 開始タスク (またはユーザー・ジョブ) を開始します。開始が成功すると、ロック・デーモンは次のコンソール・メッセージを発行します。

```
FEK501I Lock daemon started, port=4036, cleanup interval=1440,  
log level=1
```

RSED、RSE デーモン

RSED 開始タスク (またはユーザー・ジョブ) を、IVP=IVP パラメーターを指定して開始します。このパラメーターを指定すると、サーバーはいくつかのインストール検査テストを行った後に終了します。それらのテストの出力は、DD STDOUT で入手できます。エラーが発生した場合は、DD STDERR でもデータを入手できます。STDOUT データは、次のサンプルのようになります。

注: 他の IVP テストを続行する前に、IVP パラメーターを指定せずに RSE デーモンを始動してください。始動が成功すると、RSE デーモンは次のコンソール・メッセージを発行します。

```
FEK002I RseDaemon started. (port=4035)  
  
RSE daemon IVP test  
  
Wed Jul 2 17:11:52 2008 UTC  
uid=8(STCRSE) gid=1(STCGROUP)  
  
RSE daemon port is 4035  
RSE configuration files located in /etc/rdz  
  
-----  
current environment variables  
-----  
@="/usr/lpp/rdz/bin/rsed.sh" @[1]="4035" @[2]="/etc/rdz"
```

```

CGI_DTCONF="/var/rdz/scldmt"
CGI_DTWORK="/var/rdz"
CGI_TRANTABLE="FEK.#CUST.LSTRANS.FILE"
CLASSPATH=".:usr/lpp/rdz/lib:usr/lpp/rdz/lib/dstore_core.jar:usr/lpp/
ERRNO="0"
HOME="/tmp"
IFS="
"
JAVA_HOME="/usr/lpp/java/J5.0"
JAVA_PROPAGATE="NO"
LANG="C"
LIBPATH=".:usr/lib:usr/lpp/java/J5.0/bin:usr/lpp/java/J5.0/bin/classi
LINENO="66"
LOGNAME="STCRSE"
MAILCHECK="600"
OLDPWD="/tmp"
OPTIND="1"
PATH=".:usr/lpp/java/J5.0/bin:usr/lpp/rdz/bin:usr/lpp/ispf/bin:bin:/
PPID="33554711"
PS1="\$ "
PS2="> "
PS3="#? "
PS4="+ "
PWD="/etc/rdz"
RANDOM="27298"
RSE_CFG="/etc/rdz"
RSE_HOME="/usr/lpp/rdz"
RSE_LIB="/usr/lpp/rdz/lib"
SECONDS="0"
SHELL="/bin/sh"
STEPLIB="NONE"
TZ="EST5EDT"
_BPX_SHAREAS="YES"
_BPX_SPAWN_SCRIPT="YES"
_CEE_DMPTARG="/tmp"
_CEE_RUNOPTS="ALL31(ON) HEAP(32M,32K,ANYWHERE,KEEP,,) TRAP(ON)"
_CMDSERV_BASE_HOME="/usr/lpp/ispf"
_CMDSERV_CONF_HOME="/etc/rdz"
_CMDSERV_WORK_HOME="/var/rdz"
_RSE_CMDSERV_OPTS="&SESSION=SPAWN"
_RSE_DAEMON_CLASS="com.ibm.etools.zos.server.RseDaemon"
_RSE_DAEMON_IVP_TEST="1"
_RSE_DAEMON_PORT="4035"
_RSE_JAVAOPTS=" -DISPF_OPTS='&SESSION=SPAWN' -DA_PLUGIN_PATH=/usr/lpp/rd
_RSE_POOL_SERVER_CLASS="com.ibm.etools.zos.server.ThreadPoolProcess"
_RSE_PWD="/tmp"
_RSE_SERVER_CLASS="org.eclipse.dstore.core.server.Server"
_RSE_SERVER_TIMEOUT="120000"
_SCLMDT_BASE_HOME="/usr/lpp/rdz"
_SCLMDT_CONF_HOME="/var/rdz/scldmt"
_SCLMDT_TRANTABLE="FEK.#CUST.LSTRANS.FILE"
_SCLMDT_WORK_HOME="/var/rdz"
_SCLM_BASE="/var/rdz/WORKAREA"
_SCLM_BWBCALL="/usr/lpp/rdz/bin/BWBCALL"
_SCLM_DWGET="/var/rdz/WORKAREA"
_SCLM_DWTRANSFER="/var/rdz/WORKAREA"
_SCLM_J2EPUT="/var/rdz/WORKAREA"

-----
java startup test...
-----
java version "1.5.0"
Java(TM) 2 Runtime Environment, Standard Edition (build pmz31dev-2008031
IBM J9 VM (build 2.3, J2RE 1.5.0 IBM J9 2.3 z/OS s390-31 j9vmmz3123-2008
J9VM - 20080314_17962_bHdSMr
JIT - 20080130_0718ifx2_r8
GC - 200802_08)

```

JCL - 20080314

TCP/IP IVP test...

Wed Jul 2 13:11:54 EDT 2008
uid=8(STCRSE) gid=1(STCGROUP)
using /etc/rdz/rsed.envvars

TCP/IP resolver configuration (z/OS UNIX search order):

Resolver Trace Initialization Complete -> 2008/07/02 13:11:54.745964

res_init Resolver values:
Global Tcp/Ip Dataset = None
Default Tcp/Ip Dataset = None
Local Tcp/Ip Dataset = /etc/resolv.conf
Translation Table = Default
UserId/JobName = STCRSE
Caller API = LE C Sockets
Caller Mode = EBCDIC
(L) DataSetPrefix = TCPIP
(L) HostName = CDFMVS08
(L) TcpIpJobName = TCPIP
(L) DomainOrigin = RALEIGH.IBM.COM
(L) NameServer = 9.42.206.2
 9.42.206.3
(L) NsPortAddr = 53 (L) ResolverTimeout = 10
(L) ResolveVia = UDP (L) ResolverUdpRetries = 1
(*) Options NDots = 1
(*) SockNoTestStor
(*) AlwaysWto = NO (L) MessageCase = MIXED
(*) LookUp = DNS LOCAL

res_init Succeeded

res_init Started: 2008/07/02 13:11:54.755363

res_init Ended: 2008/07/02 13:11:54.755371

MVS TCP/IP NETSTAT CS V1R9 TCPIP Name: TCPIP 13:11:54

Tcpip started at 01:28:36 on 06/23/2008 with IPv6 enabled

host IP address:

hostName=CDFMVS08
hostAddr=9.42.112.75
bindAddr=9.42.112.75
localAddr=9.42.112.75

Success, addresses match

PassTicket IVP test...

Success, PassTicket IVP finished normally

RSE daemon IVP ended

サービスの検査

Developer for System z インストールは、基本およびオプションのサービス用に、いくつかのインストール検査プログラム (IVP) を提供します。IVP スクリプトは、インストール・ディレクトリー (デフォルトでは /usr/lpp/rdz/bin/) に置かれます。

表 17. サービス用の IVP

fekfivpa	126 ページの『(オプション) APPC を使用した TSO コマンド・サービス接続』
fekfivpd	123 ページの『RSE デーモン接続』
fekfivpi	124 ページの『ISPF の TSO/ISPF クライアント・ゲートウェイ接続』
fekfivpj	123 ページの『JES ジョブ・モニター接続』
fekfivpl	124 ページの『ロック・デーモン接続』
fekfivpr	128 ページの『(オプション) REXEC 接続』
fekfivps	126 ページの『(オプション) SCLMDT 接続』
fekfivpt	122 ページの『TCP/IP のセットアップ』
fekfivpz	129 ページの『(オプション) REXEC/SSH シェル・スクリプト』

以下に述べるタスクでは、ユーザーが z/OS UNIX 内でアクティブであることを想定しています。そのためには、TSO コマンド **OMVS** を発行します。TSO に戻るには、**exit** コマンドを使用します。

IVP を実行するユーザー ID には大きな領域サイズが必要です。これは、大量のメモリを必要とする機能 (Java など) が実行されるからです。領域サイズは、131072 キロバイト (128 メガバイト) 以上に設定してください。

次に示すサンプルのエラーは、領域サイズが不十分であることを明らかに示しています (ただし、ほかのエラーが発生する可能性もあります。例えば Java を始動できない場合があります)。

```
CEE5213S The signal SIGPIPE was received.
%z/OS UNIX command%: command was killed by signal number 13
  %line-number% *-* %REXX command%
    +++ RC(137) +++
```

注: IVP テストを開始する前に、Developer for System z 開始タスクをアクティブにしておく必要があります。

IVP の初期化

このセクションのサンプル・コマンドはすべて、特定の環境変数が設定されていることを想定しています。これにより、IVP スクリプトは PATH ステートメントを通じて入手でき、カスタマイズされた構成ファイルのロケーションが判明します。

pwd コマンドおよび **cd** コマンドを使用して、現行ディレクトリーを検査し、カスタマイズした構成ファイルがあるディレクトリーに変更してください。その後、次のサンプルのように ivpinit シェル・スクリプトを使用して RSE 環境変数を設定できます (\$ は z/OS UNIX プロンプトです)。

```
$ pwd
/u/userid
$ cd /etc/rdz
$ ./ivpinit
RSE configuration files located in /etc/rdz --default
added /usr/lpp/rdz/bin to PATH
```

./ivpinit の最初の「.」（ドット）は、シェルを現行環境で実行するための z/OS UNIX コマンドです。これにより、シェル内で設定された環境変数が、シェルを出た後も有効になります。2 番目のドットは現行ディレクトリーを参照しています。

注:

- ./ivpinit が fekfivp* スクリプトの前に実行されない場合は、次のサンプルのように、これらのスクリプトへのパスを呼び出し時に指定する必要があります。

```
/usr/lpp/rdz/bin/fekfivpr 512 USERID
```

また、./ivpinit が最初に実行されなかった場合、ほとんどの fekfivp* スクリプトは、カスタマイズされた rsed.envvars のロケーションの指定を要求します。

- 一部の IVP テストでは、TCP/IP REXX ソケット API が使用されます。その場合、TCP/IP ロード・ライブラリー（デフォルトは TCPIP.SEZALOAD）が LINKLIST または STEPLIB に入っている必要があります。これらの IVP テストを実行するには、以下のコマンドが必要になる場合があります（\$ は z/OS UNIX プロンプトです）。

```
$ EXPORT STEPLIB=$STEPLIB:TCPIP.SEZALOAD
```

既存の STEPLIB に APF 許可のないライブラリーを追加すると、既存の STEPLIB データ・セットの APF 許可が除去されることに注意してください。

また、CEE.SCEELKED が LINKLIST または STEPLIB 内にある場合は、TCPIP.SEZALOAD を CEE.SCEELKED の前に配置する必要があることにも注意してください。そうしないと、TCP/IP REXX ソケット呼び出しで 0C1 システム異常終了が発生します。

RSE 接続問題の診断方法については、145 ページの『第 9 章 構成問題のトラブルシューティング』、または Developer for System z サポート・ページ (<http://www-306.ibm.com/software/awdtools/rdz/support/>) の技術情報を参照してください。

ポート可用性

netstat コマンドを発行することにより、JES ジョブ・モニター、RSE デーモン、およびオプションとして REXEC または SSH ポートの可用性を検査できます。結果として、それらのサービスによって使用されているポートが次のサンプルのように表示されます（\$ は z/OS UNIX プロンプトです）。

IPv4

```
$ netstat
MVS TCP/IP NETSTAT CS VxRy    TCPIP Name: TCPIP    13:57:36
User Id Conn    Local Socket    Foreign Socket    State
```

```

-----
RSED      0000004B 0.0.0.0..4035    0.0.0.0..0    Listen
LOCKD     0000004C 0.0.0.0..4036    0.0.0.0..0    Listen
JMON      00000037 0.0.0.0..6715    0.0.0.0..0    Listen

```

IPv6

```

$ netstat
MVS TCP/IP NETSTAT CS VxRy    TCPIP Name: TCPIP    14:03:35
User Id Conn      State
-----
RSED      0000004B Listen
  Local Socket: 0.0.0.0..4035
  Foreign Socket: 0.0.0.0..0
LOCKD     0000004C Listen
  Local Socket: 0.0.0.0..4036
  Foreign Socket: 0.0.0.0..0
JMON      00000037 Listen
  Local Socket: 0.0.0.0..6715
  Foreign Socket: 0.0.0.0..0

```

TCP/IP のセットアップ

TSO コマンド・サービスに APPC を使用する場合、Developer for System z は TCP/IP が初期化時に正しいホスト名を持っているかどうかに依存します。つまり、各種の TCP/IP 構成ファイルやリゾルバー構成ファイルが正しくセットアップされていなければなりません。TCP/IP およびリゾルバーのセットアップの詳細については、329 ページの『付録 B. TCP/IP のセットアップ』を参照してください。次のコマンドを実行することにより、現行の設定値を検査します。

```
fekfivpt
```

注: この IVP では TCPIP **netstat** コマンドが発行されますが、ご使用のセキュリティー・ソフトウェアでこのコマンドが実行されないように保護されている場合があります。

このコマンドは次のサンプルのような出力を返します (\$ は z/OS UNIX プロンプトです)。

```
$ fekfivpt
```

```

Wed Jul  2 13:11:54 EDT 2008
uid=1(USERID) gid=0(GROUP)
using /etc/rdz/rsed.envvars

```

```

current address space size limit is 1914675200 (1826.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)

```

```

-----
TCP/IP resolver configuration (z/OS UNIX search order):
-----

```

```
Resolver Trace Initialization Complete -> 2008/07/02 13:11:54.745964
```

```

res_init Resolver values:
Global Tcp/Ip Dataset = None
Default Tcp/Ip Dataset = None
Local Tcp/Ip Dataset  = /etc/resolv.conf
Translation Table      = Default
UserId/JobName         = USERID
Caller API             = LE C Sockets
Caller Mode            = EBCDIC
(L) DataSetPrefix     = TCPIP
(L) HostName          = CDFMVS08

```



```

(L) TcpIpJobName = TCPIP
(L) DomainOrigin = RALEIGH.IBM.COM
(L) NameServer   = 9.42.206.2
                  9.42.206.3
(L) NsPortAddr   = 53           (L) ResolverTimeout    = 10
(L) ResolveVia   = UDP         (L) ResolverUdpRetries = 1
(*) Options NDots = 1
(*) SockNoTestStor
(*) AlwaysWto    = NO          (L) MessageCase       = MIXED
(*) LookUp       = DNS LOCAL
res_init Succeeded
res_init Started: 2008/07/02 13:11:54.755363
res_init Ended: 2008/07/02 13:11:54.755371
*****
MVS TCP/IP NETSTAT CS V1R9      TCPIP Name: TCPIP      13:11:54
Tcpip started at 01:28:36 on 06/23/2008 with IPv6 enabled

-----
host IP address:
-----

hostName=CDFMVS08
hostAddr=9.42.112.75
bindAddr=9.42.112.75
localAddr=9.42.112.75

Success, addresses match

```

RSE デーモン接続

次のコマンドを実行することにより、RSE デーモン接続を検査します。4035 は RSE デーモンが使用するポートに、また、USERID は有効なユーザー ID に置き換えてください。

```
fekfivpd 4035 USERID
```

パスワードを求めるプロンプトの後、このコマンドは次のサンプルのような出力を返します (\$ は z/OS UNIX プロンプトです)。

```

$ fekfivpd 4035 USERID

Wed Jul  2 15:00:27 EDT 2008
uid=1(USERID) gid=0(GROUP)
using /etc/rdz/rsed.envvars

current address space size limit is 1914675200 (1826.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)

Password:
SSL is disabled
connected
8108
570655399
Success

```

注: SSL 使用可能接続をテストしたとき、「gsk_secure_socket_init() failed: Socket closed by remote partner」というエラー・メッセージを受け取った場合は、正しいポートを指定したかどうか、検査してください。

JES ジョブ・モニター接続

次のコマンドを実行することにより、JES ジョブ・モニター接続を検査します。6715 は JES ジョブ・モニターのポート番号に置き換えてください。

```
fekfivpj 6715
```

このコマンドは、次のサンプルに示すような JES ジョブ・モニター確認応答メッセージを返します (\$ は z/OS UNIX プロンプトです)。

```
$ fekfivpj 6715
```

```
Wed Jul 2 15:00:27 EDT 2008
uid=1(USERID) gid=0(GROUP)
using /etc/rdz/rsed.envvars
```

```
current address space size limit is 1914675200 (1826.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)
```

```
hostName=CDFMVS08
hostAddr=9.42.112.75
Waiting for JES Job Monitor response...
ACKNOWLEDGE01v03
```

```
Success
```

ロック・デーモン接続

次のコマンドを実行することにより、ロック・デーモン接続を検査します。

```
fekfivpl
```

このコマンドは次のサンプルのような出力を返します (\$ は z/OS UNIX プロンプトです)。

```
$ fekfivpl
```

```
Mon Jun 29 08:00:38 EDT 2009
uid=1(USERID) gid=0(GROUP)
using /etc/rdz/rsed.envvars
```

```
current address space size limit is 1914675200 (1826.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)
```

```
hostName=CDFMVS08
hostAddr=9.42.112.75
```

```
Registering user to Lock Daemon...
Waiting for Lock Daemon response...
```

```
Querying to Lock Daemon...
Waiting for Lock Daemon response...
USERID
```

```
Unregistering user from Lock Daemon...
Waiting for Lock Daemon response...
```

```
Querying to Lock Daemon...
Waiting for Lock Daemon response...
```

```
Success
```

ISPF の TSO/ISPF クライアント・ゲートウェイ接続

次のコマンドを実行することにより、ISPF の TSO/ISPF クライアント・ゲートウェイへの接続を検査します。

```
fekfivpi
```

このコマンドは、次のサンプルに示すような、ISPF の TSO/ISPF クライアント・ゲートウェイに関連した検査 (変数、HFS モジュール、TSO/ISPF セッションの開始および停止) の結果を返します (\$ は z/OS UNIX プロンプトです)。

```
$ fekfivpi
```

```
Wed Jul 2 15:00:27 EDT 2008
uid=1(USERID) gid=0(GROUP)
using /etc/rdz/rsed.envvars
```

```
current address space size limit is 1914675200 (1826.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)
```

```
-----
/etc/rdz/ISPF.conf content:
-----
```

```
isplib=ISP.SISPMENU
isptlib=ISP.SISPTENU
ispplib=ISP.SISPPENU
ispslib=ISP.SISPSLIB
sysproc=ISP.SISPLIB,FEK.SFEKPROC
```

```
-----
Host install verification for RSE
Review IVP log messages from HOST below :
-----
```

```
RSE connection and base TSO/ISPF session initialization check only
```

```
*** CHECK : ENVIRONMENT VARIABLES - key variables displayed below :
```

```
Server PATH          =
/usr/lpp/java/J5.0/bin:/usr/lpp/rdz/lib:/usr/lpp/ispf/bin:
/bin:/usr/sbin:.
```

```
STEPLIB              = FEK.SFEKAUTH:FEK.SFEKLOAD
```

```
_CMDSERV_BASE_HOME   = /usr/lpp/ispf
_CMDSERV_CONF_HOME    = /etc/rdz
_CMDSERV_WORK_HOME    = /var/rdz
-----
```

```
*** CHECK : USS MODULES
Checking ISPF Directory : /usr/lpp/ispf
Checking modules in /usr/lpp/ispf/bin directory
Checking for ISPF configuration file ISPF.conf
RC=0
MSG: SUCCESSFUL
```

```
-----
*** CHECK : TSO/ISPF INITIALIZATION
( TSO/ISPF session will be initialized )
RC=0
MSG: SUCCESSFUL
```

```
-----
*** CHECK: Shutting down TSO/ISPF IVP session
RC=0
MSG: SUCCESSFUL
```

```
-----
Host installation verification completed successfully
-----
```

注: いずれかの ISPF 検査が失敗した場合は、さらに詳細な情報が表示されます。

fekfivpi には、以下に示すオプションの非定位置パラメーターがあります。

- file** fekfivpi は、大量 (数百行) の出力を生成する場合があります。-file パラメーターは、この出力をファイル userlog/.eclipse/RSE/\$LOGNAME/fekfivpi.log へ送信します。ここで、userlog は rsed.envvars 内の user.log ディレクティブの値で、\$LOGNAME はユーザー ID (大文字) です。user.log ディレクティブがコメント化されているか存在しない場合は、ホーム・パスが使用されます。ホーム・パスは、OMVS セキュリティー・セグメントで定義されます。
- debug** -debug パラメーターは、詳細なテスト出力を作成します。IBM サポートから指示された場合以外は、このオプションを使用しないでください。

(オプション) APPC を使用した TSO コマンド・サービス接続

次のコマンドを実行することにより、APPC を使用する TSO コマンド・サーバーへの接続を検査します。USERID は、有効なユーザー ID に置き換えてください。

```
fekfivpa USERID
```

パスワードを求めるプロンプトの後、このコマンドは次のサンプルのような APPC 会話を返します (\$ は z/OS UNIX プロンプトです)。

```
$ fekfivpa USERID
Enter password:
```

```
Wed Jul  2 15:00:27 EDT 2008
uid=1(USERID) gid=0(GROUP)
using /etc/rdz/rsed.envvars
```

```
current address space size limit is 1914675200 (1826.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)
```

```
20070607 13:57:18.584060 /usr/lpp/rdz/bin/fekfscmd: version=0ct 2003
20070607 13:57:18.584326 Input parms: 1.2.3.4 * NOTRACE USERID *****
20070607 13:57:18.586800 APPC: Allocate succeeded
20070607 13:57:18.587022 Conversation id is 0DDBD3F80000000D
20070607 13:57:18.587380 APPC: Set Send Type succeeded
20070607 13:57:26.736674 APPC: Confirm succeeded
20070607 13:57:26.737027 Req to send recd value is 0
20070607 13:57:26.737546 APPC: SEND_DATA return_code = 0
20070607 13:57:26.737726 request_to_send_received = 0
20070607 13:57:26.737893 Send Data succeeded
20070607 13:57:26.738169 APPC: Set Prepare to Receive type succeeded
20070607 13:57:26.738580 APPC: Prepare to Receive succeeded
20070607 13:57:26.808899 APPC: Receive data
20070607 13:57:26.809122 RCV return_code = 0
20070607 13:57:26.809270 RCV data_received= 2
20070607 13:57:26.809415 RCV received_length= 29
20070607 13:57:26.809556 RCV status_received= 4
20070607 13:57:26.809712 RCV req_to_send= 0
20070607 13:57:26.809868 Receive succeeded
:IP: 0 9.42.112.75 1674 50246
20070607 13:57:26.810533 APPC: CONFIRMED succeeded
```

(オプション) SCLMDT 接続

次のコマンドを実行することにより、SCLM Developer Toolkit への接続を検査します。

```
fekfivps
```

このコマンドは、次のサンプルに示すような SCLM Developer Toolkit 関連の検査 (変数、HFS モジュール、REXX ランタイム、TSO/ISPF セッションの開始および停止) の結果を返します (\$ は z/OS UNIX プロンプトです)。

```
$ fekfivps
```

```
Wed Jul 2 15:00:27 EDT 2008
uid=1(USERID) gid=0(GROUP)
using /etc/rdz/rsed.envvars
```

```
current address space size limit is 1914675200 (1826.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)
```

```
-----
/etc/rdz/ISPF.conf content:
-----
```

```
isplib=ISP.SISPMENU
isptlib=ISP.SISPTENU
ispplib=ISP.SISPPENU
ispslib=ISP.SISPSLIB
sysproc=ISP.SISPCLIB,FEK.SFEKPROC
-----
```

```
Host install verification for RSE
Review IVP log messages from HOST below :
-----
```

```
*** CHECK : ENVIRONMENT VARIABLES - key variables displayed below :
```

```
Server PATH          = /usr/lpp/java/J5.0/bin:/usr/lpp/rdz/lib:/usr/lpp/ispf/bin:
/bin:/usr/sbin:.
```

```
STEPLIB              = FEK.SFEKAUTH:FEK.SFEKLOAD
```

```
_CMDSERV_BASE_HOME   = /usr/lpp/ispf
_CMDSERV_CONF_HOME   = /etc/rdz
_CMDSERV_WORK_HOME    = /var/rdz
_SCLMDT_CONF_HOME     = /var/rdz/sclmdt
_SCLMDT_WORK_HOME     = /var/rdz
_SCLMDT_TRANTABLE     = FEK.#CUST.LSTRANS.FILE
-----
```

```
*** CHECK : JAVA PATH SETUP VERIFICATION
RC=0
MSG: SUCCESSFUL
-----
```

```
*** CHECK : USS MODULES
Checking ISPF Directory : /usr/lpp/ispf
Checking modules in /usr/lpp/ispf/bin directory
Checking for ISPF configuration file ISPF.conf
Checking install bin Directory : /usr/lpp/rdz/bin
RC=0
MSG: SUCCESSFUL
-----
```

```
*** CHECK : REXX RUNTIME ENVIRONMENT
RC=0
MSG: SUCCESSFUL
-----
```

```
*** CHECK : TSO/ISPF INITIALIZATION
( TSO/ISPF session will be initialized )
RC=0
MSG: SUCCESSFUL
-----
```

```
*** CHECK: Shutting down TSO/ISPF IVP session
RC=0
MSG: SUCCESSFUL
```

```
-----
Host installation verification completed successfully
-----
```

注: いずれかの SCLMDT 検査が失敗した場合は、さらに詳細な情報が表示されます。

fekfivps には、以下に示すオプションの非定位置パラメーターがあります。

-file fekfivps は、大量 (数百行) の出力を生成する場合があります。-file パラメーターは、この出力をファイル userlog/.eclipse/RSE/\$LOGNAME/fekfivps.log へ送信します。ここで、userlog は rsed.envvars 内の user.log ディレクティブの値で、\$LOGNAME はユーザー ID (大文字) です。user.log ディレクティブがコメント化されているか存在しない場合は、ホーム・パスが使用されます。ホーム・パスは、OMVS セキュリティー・セグメントで定義されます。

-debug -debug パラメーターは、詳細なテスト出力を作成します。IBM サポートから指示された場合以外は、このオプションを使用しないでください。

(オプション) REXEC 接続

次のコマンドを実行することにより、REXEC 接続を検査します。512 は REXEC が使用するポートに、また、USERID は有効なユーザー ID に置き換えてください。

```
fekfivpr 512 USERID
```

パスワードを求めるプロンプトの後、このコマンドは REXEC トレース、タイムアウトの警告、Java バージョン、および RSE サーバー・メッセージを次のサンプルのように返します (\$ は z/OS UNIX プロンプトです)。

```
$ fekfivpr 512 USERID
Enter password:
```

```
Wed Jul  2 15:00:27 EDT 2008
uid=1(USERID) gid=0(GROUP)
using /etc/rdz/rsed.envvars
```

```
current address space size limit is 1914675200 (1826.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)
```

```
$ EZYRC01I Calling function rexec_af with the following:
EZYRC02I Host: CDFMVS08, user USERID, cmd cd /etc/rdz;export RSE_USER_ID=USERI
D;./server.zseries -ivp, port 512
EZYRC19I Data socket = 4, Control socket = 6.
```

```
RSE server IVP test
```

```
CDFMVS08 -- Wed Jul  2 15:00:27 EDT 2008
uid=1(USERID) gid=0(GROUP)
```

```
RSE configuration files located in /etc/rdz --default
```

```
RSE userid is USERID --default
```

```
-----
Address Space size limits
-----
```

```
current address space size limit is 2147483647 (2048.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)
```

```
-----
service history
-----
```

```
Fri Jun 19 00:01:00 2009 -- COPY -- HHOP760 v7600 created 18 Jun 2009
```

```
-----
expect to see time out messages after a successful IVP test
-----
```

```
-----
starting RSE server in background -- Fri Jun 19 15:59:05 EDT 2009
-----
```

```
java version "1.5.0"
Java(TM) 2 Runtime Environment, Standard Edition (build pmz31dev-20070201 (SR4))
IBM J9 VM (build 2.3, J2RE 1.5.0 IBM J9 2.3 z/OS s390-31 j9vmz3123-20070201 (JIT
enabled)
J9VM - 20070131_11312_bHdSMr
JIT - 20070109_1805ifx1_r8
GC - 200701_09)
JCL - 20070126
```

```
DStore Server Starting...
Server Started Successfully
8108
Server running on: CDFMVS08
```

注:

- Java および RSE サーバーの出力を取得できない場合は、INETD の領域サイズが小さすぎる可能性があります (このサイズは、TSO/OMVS シェル・セッションから始動した場合は 2096128 以上でなければならず、BPXBATCH の場合は領域サイズ 0 でなければなりません)。
- REXEC によって使用されるシェル・スクリプトは、次の IVP テスト (『(オプション) REXEC/SSH シェル・スクリプト』) で説明されているように、別個にテストすることができます。
- サーバーは、接続を試みるクライアントなしに始動されるため、(5 秒後に) タイムアウトになります。その結果、次のサンプルに示すような接続エラー・メッセージが生成されます。

```
Connection error
Server: error initializing socket: java.net.SocketTimeoutException:
Accept timed out
```

(オプション) REXEC/SSH シェル・スクリプト

この IVP テストは、128 ページの『(オプション) REXEC 接続』で概説した前のテストが正常に完了した場合はスキップすることができます。

次のコマンドを実行することにより、REXEC および SSH 接続で使用されるシェル・スクリプトを検査します。

```
fekfivpz
```

このコマンドは、タイムアウトの警告、Java バージョン、および RSE サーバー・メッセージを次のサンプルのように返します (\$ は z/OS UNIX プロンプトです)。

```
$ fekfivpz
```

```
Wed Jul 2 15:00:27 EDT 2008
```



```

uid=1(USERID) gid=0(GROUP)

using /etc/rdz/rsed.envvars

current address space size limit is 1914675200 (1826.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)

-----

RSE server IVP test

CDFMVS08 -- Wed Jul  2 15:00:27 EDT 2008
uid=1(USERID) gid=0(GROUP)

RSE configuration files located in /etc/rdz --default
RSE userid is USERID --default

-----

Address Space size limits
-----
current address space size limit is 2147483647 (2048.0 MB)
maximum address space size limit is 2147483647 (2048.0 MB)

-----

service history
-----
Fri Jun 19 00:01:00 2009 -- COPY -- HHOP760 v7600 created 18 Jun 2009

-----

expect to see time out messages after a successful IVP test

-----

starting RSE server in background -- Fri Jun 19 15:59:05 EDT 2009
-----
java version "1.5.0"
Java(TM) 2 Runtime Environment, Standard Edition (build pmz31dev-20070201 (SR4))
IBM J9 VM (build 2.3, J2RE 1.5.0 IBM J9 2.3 z/OS s390-31 j9vmmz3123-20070201 (JIT enabled))
J9VM - 20070131_11312_bHdSMr
JIT  - 20070109_1805ifx1_r8
GC   - 200701_09)
JCL  - 20070126

DStore Server Starting...
Server Started Successfully
8108
Server running on: CDFMVS08

```

注:

- 出力を取得できない場合は、使用している (TSO) 領域サイズが小さすぎる可能性があります (このサイズは 2096128 でなければなりません)。
- サーバーは、接続を試みるクライアントなしに始動されるため、(5 秒後に) タイムアウトになります。その結果、次のサンプルに示すような接続エラー・メッセージが生成されます。

```

Connection error
Server: error initializing socket: java.net.SocketTimeoutException:
        Accept timed out

```

第 2 部 Developer for System z 情報

第 8 章 オペレーター・コマンド

この章では、Developer for System z で使用可能なオペレーター（またはコンソール）コマンドの概要を説明します。コマンド・フォーマットの説明に使用される構文図がよく分からない場合は、143 ページの『構文図の読み方』を参照してください。

Start (S)

START コマンドは、開始タスク (STC) を動的に開始するために使用します。このコマンドの省略形バージョンは英字の **S** です。

JES ジョブ・モニター

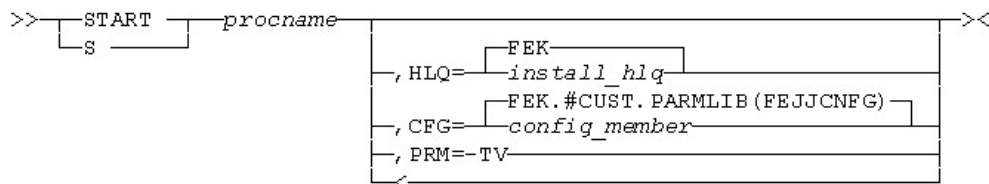


図 33. *START JMON* オペレーター・コマンド

procname

サーバーを始動するために使用する、プロシージャ・ライブラリー内のメンバーの名前。ホスト構成時に使用されるデフォルト名は **JMON** です。

HLQ=install_hlq

Developer for System z をインストールするために使用する高位修飾子。デフォルトは **FEK** です。

CFG=config_member

JES ジョブ・モニター構成ファイルの絶対データ・セットおよびメンバー名。デフォルトは **FEK.#CUST.PARMLIB (FEJJCNFG)** です。

PRM=-TV

冗長（トレース）モードを使用可能にします。トレースはパフォーマンス低下の原因になるため、IBM サポートの指示の下でのみ実行してください。

RSE デーモン

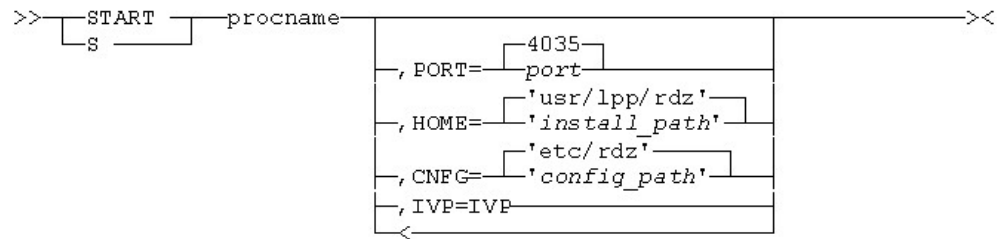


図 34. START RSED オペレーター・コマンド

procname

サーバーを始動するために使用する、プロシージャー・ライブラリー内のメンバーの名前。ホスト構成時に使用されるデフォルト名は RSED です。

PORT=port

RSE デーモンがクライアントの接続に使用するポート。デフォルトは 4035 です。

HOME='install_path'

Developer for System z をインストールするために使用するパス接頭部および必須の /usr/lpp/rdz。デフォルトは '/usr/lpp/rdz' です。z/OS UNIX パスには大/小文字の区別があり、小文字を保存するには単一引用符 (') で囲む必要があることに注意してください。

CNFG='config_path'

z/OS UNIX 内に保管される構成ファイルの絶対ロケーション。デフォルトは '/etc/rdz' です。z/OS UNIX パスには大/小文字の区別があり、小文字を保存するには単一引用符 (') で囲む必要があることに注意してください。

IVP=IVP

サーバーを始動しませんが、RSE デーモン・インストール検査プログラム (IVP) を実行します。

ロック・デーモン

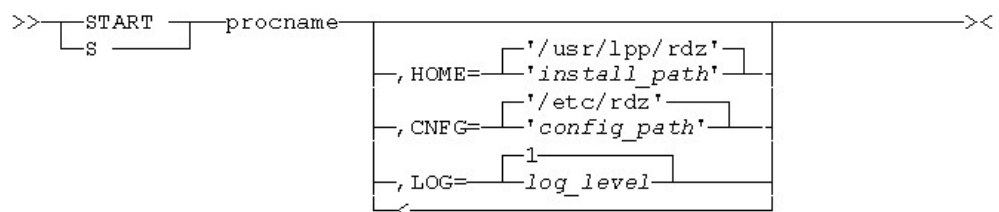


図 35. START LOCKD オペレーター・コマンド

procname

サーバーを始動するために使用する、プロシージャー・ライブラリー内のメンバーの名前。ホスト構成時に使用されるデフォルト名は LOCKD です。

HOME='install_path'

Developer for System z をインストールするために使用するパス接頭部および必須の /usr/lpp/rdz。デフォルトは '/usr/lpp/rdz' です。z/OS UNIX パスには大/小文字の区別があり、小文字を保存するには単一引用符 (') で囲む必要があることに注意してください。

CNFG='config_path'

z/OS UNIX 内に保管される構成ファイルの絶対ロケーション。デフォルトは '/etc/rdz' です。z/OS UNIX パスには大/小文字の区別があり、小文字を保存するには単一引用符 (') で囲む必要があることに注意してください。

LOG=log_level

DD STDOUT の出力の詳細レベル。

- 0 : エラー・メッセージのみをログに記録します。
- 1 : エラー・メッセージと警告メッセージをログに記録します (デフォルト)。
- 2 : エラー・メッセージ、警告メッセージ、および情報メッセージをログに記録します。

Modify (F)

MODIFY コマンドを使用すると、アクティブ・タスクの特性を動的に照会および変更することができます。このコマンドの省略形バージョンは英字の **F** です。

JES ジョブ・モニター

```
>> [MODIFY] [procname] [, APPL=-TV] _____><
      [F]      [, APPL=-TN]
```

図 36. **MODIFY JMON** オペレーター・コマンド

procname

サーバーを始動するために使用された、プロシージャ・ライブラリー内のメンバーの名前。ホスト構成時に使用されるデフォルト名は **JMON** です。

-TV 冗長 (トレース) モードを使用可能にします。トレースはパフォーマンス低下の原因になるため、IBM サポートの指示の下でのみ実行してください。

-TN 冗長 (トレース) モードを使用不可にします。

RSE デモン

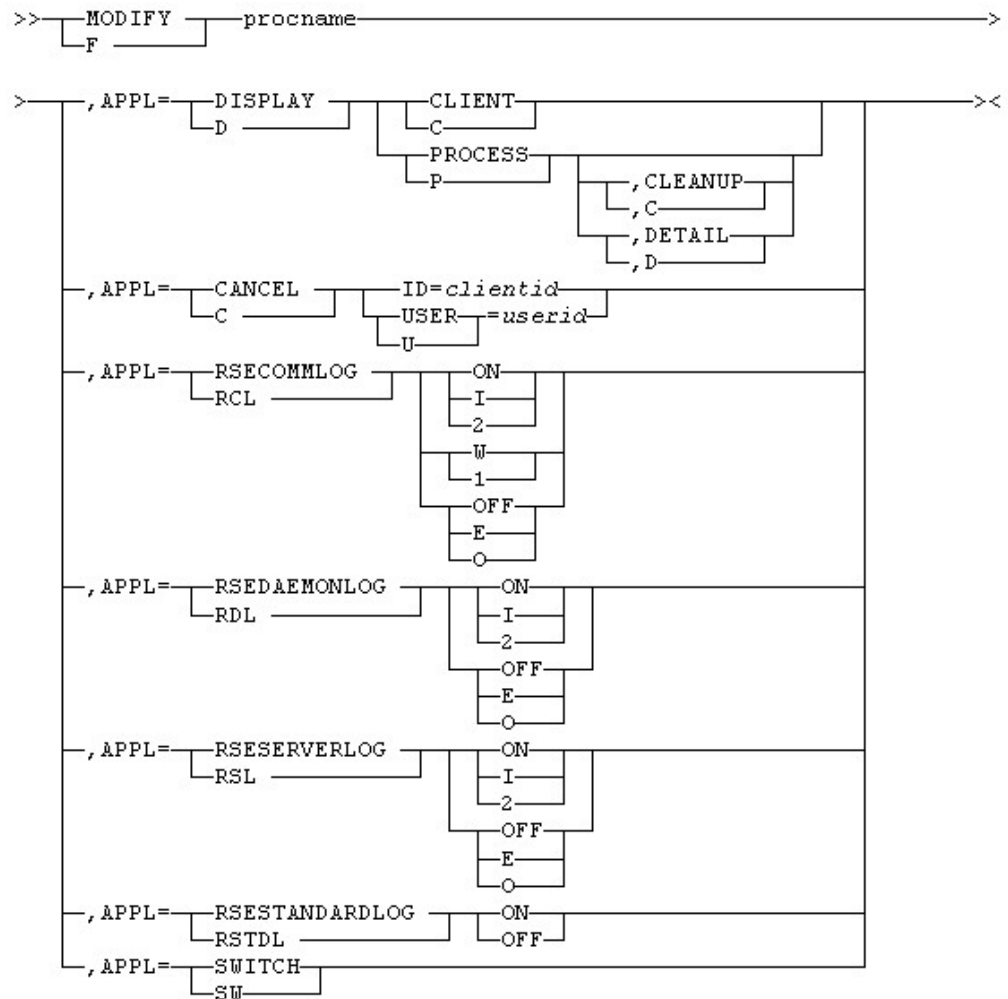


図 37. MODIFY RSED オペレーター・コマンド

procname

サーバーを始動するために使用された、プロシージャ・ライブラリー内のメンバーの名前。ホスト構成時に使用されるデフォルト名は RSED です。

DISPLAY CLIENT

アクティブなクライアントを表示します。

<clientid> : <userid> : <connected since>

DISPLAY PROCESS[CLEANUP,DETAIL]

RSE スレッド・プール・プロセスを表示します。接続したユーザーのロード・バランシングに複数のプロセスが使用されている可能性があります。

ProcessId(<processid>) Memory Usage(<java heap usage>%)
Clients(<number of clients>) Order(<startup order>) <error status>

注:

- <processid> は、プロセス固有の z/OS UNIX オペレーター・コマンドで使用できます。

- 各プロセスには独自の Java ヒープがあり、そのサイズは `rsed.envvars` の中で設定できます。
- `<startup order>` は、スレッド・プールが開始された順序を示す連番です。この番号は、`stderr.*.log` および `stdout.*.log` ファイルのファイル名に使用されている数字と一致します。

通常の状態では、`<error status>` はブランクです。表 18 に、`<error status>` のブランク以外の値を示します。

表 18. スレッド・プールのエラー状況

状況	説明
severe error	スレッド・プール・プロセスでリカバリー不能エラーが発生し、操作が停止されました。その他の状況フィールドには、最後に認識された値が示されます。この項目をテーブルから除去するには、 DISPLAY PROCESS 変更コマンドの CLEANUP オプションを使用します。
killed process	スレッド・プール・プロセスが、Java、z/OS UNIX、またはオペレーター・コマンドによって強制終了されました。その他の状況フィールドには、最後に認識された値が示されます。この項目をテーブルから除去するには、 DISPLAY PROCESS 変更コマンドの CLEANUP オプションを使用します。
timeout	クライアント接続要求で、スレッド・プール・プロセスが時間内に RSE デーモンに応答しませんでした。その他の状況フィールドには、現行値が示されます。このスレッド・プールは、今後のクライアント接続要求で除外されます。 *timeout* 状況は、このスレッド・プールで処理されているクライアントがログオフするとリセットされます。

DISPLAY PROCESS 変更コマンドの **DETAIL** オプションを使用すると、詳細情報が表示されます。

```
ProcessId(33555087) ASId(002E) JobName(RSED8) Order(1)
PROCESS LIMITS:  CURRENT  HIGHWATER    LIMIT
JAVA HEAP USAGE(%)      10         56         100
CLIENTS                  0          25          60
MAXFILEPROC              83         103       64000
MAXPROCUSER              97          99        200
MAXTHREADS                9          14       1500
MAXTHREADTASKS           9          14       1500
```

ASId フィールドは、16 進表記のアドレス・スペース ID です。
「**PROCESS LIMITS**」の表には、現在のリソース使用量、リソース使用量の最高水準点、およびリソースの限度が示されます。他の制限要因のために、定義された限度に到達しない場合があることに注意してください。

CANCEL ID=clientid

クライアント ID に基づいて、クライアント接続をキャンセルします。クライアント ID は **DISPLAY CLIENT** 変更コマンドで表示されます。

CANCEL USER=userid

クライアントのユーザー ID に基づいて、クライアント接続をキャンセルします。クライアントのユーザー ID は **DISPLAY CLIENT** 変更コマンドで表示されます。

RSECOMMLOG {ON,OFF,I,W,E,2,1,0}

RSE サーバー (rsecomm.log) および MVS データ・セット・サービス (lock.log および ffs*.log) のトレース詳細レベルを制御します。始動デフォルトは rsecomm.properties で定義されます。以下の 3 つの詳細レベルを使用できます。

E または 0 または OFF	エラー・メッセージのみ。
W または 1	エラー・メッセージと警告メッセージ。これは、rsecomm.properties でのデフォルトの設定です。
I または 2 または ON	エラー・メッセージ、警告メッセージ、および情報メッセージ。

詳細なトレースはパフォーマンス低下の原因になるため、IBM サポートの指示の下でのみ実行してください。

RSEDAEMONLOG {ON,OFF,I,E,2,0}

RSE デーモン (rsedaemon.log) のトレース詳細レベルを制御します。始動デフォルトは rsecomm.properties で定義されます。以下の 2 つの詳細レベルを使用できます。

E または 0 または OFF	エラー・メッセージのみ。
I または 2 または ON	エラー・メッセージ、警告メッセージ、および情報メッセージ。

詳細なトレースはパフォーマンス低下の原因になるため、IBM サポートの指示の下でのみ実行してください。

RSESERVERLOG {ON,OFF,I,E,2,0}

RSE スレッド・プール (rseserver.log) のトレース詳細レベルを制御します。始動デフォルトは rsecomm.properties で定義されます。以下の 2 つの詳細レベルを使用できます。

E または 0 または OFF	エラー・メッセージのみ。
I または 2 または ON	エラー・メッセージ、警告メッセージ、および情報メッセージ。

詳細なトレースはパフォーマンス低下の原因になるため、IBM サポートの指示の下でのみ実行してください。

RSESTANDARDLOG {ON,OFF}

スレッド・プールの stdout および stderr ストリームを保持しているログ・ファイル (stdout*.log と stderr*.log) の更新を、無効 (OFF) または有

効 (ON) にします。始動デフォルトは、rsed.envvars 内の enable.standard.log ディレクティブで定義されます。

詳細なトレースはパフォーマンス低下の原因になるため、IBM サポートの指示の下でのみ実行してください。

SWITCH

新しい監査ログ・ファイルに切り替えます。

注:

- 上記のログ・ファイルの詳細については、145 ページの『第 9 章 構成問題のトラブルシューティング』の 146 ページの『ログ・ファイル』を参照してください。
- 監査の詳細については、169 ページの『第 10 章 セキュリティーに関する考慮事項』の 176 ページの『監査ロギング』を参照してください。

ロック・デーモン

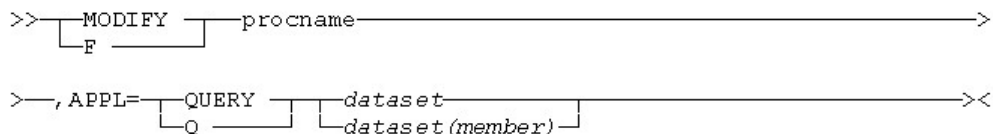


図 38. MODIFY LOCKD オペレーター・コマンド

procname

サーバーを始動するために使用された、プロシージャ・ライブラリー内のメンバーの名前。ホスト構成時に使用されるデフォルト名は LOCKD です。

QUERY dataset[(member)]

リストされたデータ・セットまたはメンバーのロック状況を照会します。サーバーは、以下のいずれかのメッセージで応答します。

```
BPXM023I (stclock) dataset[(member)] NOT LOCKED
BPXM023I (stclock) dataset[(member)] LOCKED BY userid
```

注:

- また、サーバーは他の製品 (ISPF など) によって保持されているロックも報告します。
- ロック・デーモンに登録できなかった Developer for System z クライアントが保持しているロックでは、スレッド・プール・サーバー・アドレス・スペース (RSEDx) がロック所有者として報告される結果になります。

RSE サーバーがロック・デーモンにクライアントを登録できない場合は、コンソール・メッセージ FEK513W が生成されます。このメッセージに記述されている ASID 値および TCB 値を **D**

GRS,RES=(*,dataset[(member)]) オペレーター・コマンドの出力と比較して、ロックを保持している実際のユーザーを知ることができます。

Stop (P)

アクティブ・タスクを停止するには、**STOP** コマンドを使用します。このコマンドの省略形バージョンは英字の **P** です。

```
>> STOP procname <<
  P
```

図 39. STOP オペレーター・コマンド

procname

サーバーを始動するために使用された、プロシージャ・ライブラリー内のメンバーの名前。ホスト構成時に使用されるデフォルト名は、JES ジョブ・モニターの場合は JMON、RSE デーモンの場合は RSED、およびロック・デーモンの場合は LOCKD です。

コンソール・メッセージ

JES ジョブ・モニター

JES ジョブ・モニターには、製品固有のコンソール・メッセージはありません。サーバーは Developer for System z クライアントが実行したアクションのコンソール・メッセージを生成する際、z/OS および JES に依存します。

RSE デーモン、RSE スレッド・プール・サーバー、およびロック・デーモン

表 19 は、RSE デーモン、RSE スレッド・プール・サーバー、およびロック・デーモンが生成する製品固有のコンソール・メッセージのリストです。

表 19. RSE コンソール・メッセージ

メッセージ ID	メッセージ・テキスト
FEK001I	RseDaemon を {0} ビット・モードで初期化します。(RseDaemon being initialized in {0} bit mode)
FEK002I	RseDaemon が始動しました。(RseDaemon started.) (ポート={0}) ((port={0}))
FEK003I	停止コマンドを処理中です。(Stop command being processed)
FEK004I	RseDaemon: 最大ヒープ・サイズ = {0}MB および専用 AS サイズ = {1}MB (Max Heap Size={0}MB and private AS Size={1}MB)
FEK005I	サーバー・プロセスが開始されました。(Server process started.) (processId={0})
FEK009I	RseDaemon がサーバー・プロセスの開始を待っています。(RseDaemon is waiting for the server process to start.)
FEK010I	(rsed.envvars ロケーション = {0}) ((rsed.envvars location = {0}))
FEK011I	(ログ・ディレクトリ = {0}) ((log directory = {0}))
FEK100E	デーモン・ポート/タイムアウト値は数字でなければなりません。(Daemon port/timeout value must be digits)
FEK101E	JRE {0} 以上が必要です。(JRE {0} or higher required)

表 19. RSE コンソール・メッセージ (続き)

メッセージ ID	メッセージ・テキスト
FEK102E	無効な引数を受け取りました: {0} (Invalid arguments received: {0})
FEK103E	{0} のディスクがほとんど満杯です。(Almost Disk-Full in {0})
FEK104E	最大処理数に到達しました。(Maximum number of processes has been reached)
FEK105E	監査データ送信中のエラー (rc={0}) (Error in sending audit data (rc={0}))
FEK110E	socket() が失敗しました。(socket() failed.) 理由={0} (reason={0}))
FEK111E	setsockopt() が失敗しました。(setsockopt() failed.) 理由={0} (reason={0}))
FEK112E	bind() が失敗しました。(bind() failed.)理由={0} (reason={0}))
FEK113E	listen() が失敗しました。(listen() failed.) 理由={0} (reason={0}))
FEK114E	accept() が失敗しました。(accept() failed.) 理由={0} (reason={0}))
FEK115E	write() が失敗しました。(write() failed.) 理由={0} (reason={0}))
FEK116E	pipe() が失敗しました。(pipe() failed.)理由={0} (reason={0}))
FEK117E	socketpair() が失敗しました。(socketpair() failed.) 理由={0} (reason={0}))
FEK118E	select() が失敗しました。(select() failed.) 理由={0} (reason={0}))
FEK119E	_console() が失敗しました。(_console() failed.)理由={0} (reason={0}))
FEK130E	gsk_environment_open() が失敗しました。(gsk_environment_open() failed.) 理由={0} (reason={0}))
FEK131E	gsk_attribute_set_enum(GSK_PROTOCOL_SSLV2) が失敗しました。(gsk_attribute_set_enum(GSK_PROTOCOL_SSLV2) failed.) 理由={0} (reason={0}))
FEK132E	gsk_attribute_set_enum(GSK_PROTOCOL_SSLV3) が失敗しました。(gsk_attribute_set_enum(GSK_PROTOCOL_SSLV3) failed.) 理由={0} (reason={0}))
FEK133E	gsk_attribute_set_enum(GSK_PROTOCOL_TLSV1) が失敗しました。(gsk_attribute_set_enum(GSK_PROTOCOL_TLSV1) failed.) 理由={0} (reason={0}))
FEK134E	gsk_attribute_set_buffer(GSK_KEYRING_FILE) が失敗しました。(gsk_attribute_set_buffer(GSK_KEYRING_FILE) failed.) 理由={0} (reason={0}))
FEK135E	gsk_attribute_set_buffer(GSK_KEYRING_PW) が失敗しました。(gsk_attribute_set_buffer(GSK_KEYRING_PW) failed.) 理由={0} (reason={0}))
FEK136E	gsk_environment_init() が失敗しました。(gsk_environment_init() failed.) 理由={0} (reason={0}))
FEK137E	gsk_secure_socket_open() が失敗しました。(gsk_secure_socket_open() failed.) 理由={0} (reason={0}))
FEK138E	gsk_attribute_set_numeric_value(GSK_FD) が失敗しました。(gsk_attribute_set_numeric_value(GSK_FD) failed.) 理由={0} (reason={0}))

表 19. RSE コンソール・メッセージ (続き)

メッセージ ID	メッセージ・テキスト
FEK139E	gsk_attribute_set_buffer(GSK_KEYRING_LABEL) が失敗しました。 (gsk_attribute_set_buffer(GSK_KEYRING_LABEL) failed.) 理由=({0}) (reason=({0}))
FEK140E	gsk_attribute_set_enum(GSK_SESSION_TYPE) が失敗しました。 (gsk_attribute_set_enum(GSK_SESSION_TYPE) failed.) 理由=({0}) (reason=({0}))
FEK141E	gsk_attribute_set_callback(GSK_IO_CALLBACK) が失敗しました。 (gsk_attribute_set_callback(GSK_IO_CALLBACK) failed.) 理由=({0}) (reason=({0}))
FEK142E	gsk_secure_socket_init() が失敗しました。(gsk_secure_socket_init() failed.) 理由=({0}) (reason=({0}))
FEK143E	gsk_attribute_set_enum(GSK_CLIENT_AUTH_TYPE) が失敗しました。 (gsk_attribute_set_enum(GSK_CLIENT_AUTH_TYPE) failed.) 理由=({0}) (reason=({0}))
FEK144E	gsk_get_cert_info が失敗しました。(gsk_get_cert_info failed.) 理由=({0}) (reason=({0}))
FEK145E	gsk_secure_socket_read() が失敗しました。(gsk_secure_socket_read() failed.) 理由=({0}) (reason=({0}))
FEK146E	gsk_secure_socket_write() が失敗しました。(gsk_secure_socket_write() failed.) 理由=({0}) (reason=({0}))
FEK150E	RseDaemon が異常終了しました。{0} (RseDaemon abnormally terminated; {0})
FEK201I	{0} コマンドが処理されました。({0} Command has been processed)
FEK202E	無効なコマンドが入力されました。(Invalid Command Entered)
FEK203E	無効な Display コマンド: Display Process Client (Invalid Display Command: Display Process Client)
FEK204E	無効な Cancel コマンド: Cancel ID= User= (Invalid Cancel Command: Cancel ID= User=)
FEK205E	コマンドは、連続した SWITCH のために処理されませんでした。 (Command was not processed owing to consecutive SWITCHs)
FEK206E	監査ログ機能がアクティブではありません。(Audit Log facility is not active)
FEK207I	表示するクライアントがありません。(No Client to be displayed)
FEK208I	{0} がキャンセルされました。({0} canceled)
FEK209I	表示するプロセスがありません。(No Process to be displayed)
FEK210I	{0} が重複ログオンのためにキャンセルされました。({0} canceled owing to duplicate logon)
FEK501I	ロック・デーモンが始動しました。ポート ={0}、クリーンアップ間隔 ={1}、ログ・レベル ={2} (Lock daemon started, port={0}, cleanup interval={1}, log level={2})
FEK502I	ロック・デーモンの終了処理中です。(Lock daemon terminating)
FEK510E	ロック・デーモン、ポートの欠落 (Lock daemon, missing port)
FEK511E	ロック・デーモン、誤ったポート、ポート ={0} (Lock daemon, wrong port, port={0})

表 19. RSE コンソール・メッセージ (続き)

メッセージ ID	メッセージ・テキスト
FEK512E	ロック・デーモン、ソケット・エラー、ポート = {0} (Lock daemon, socket error, port={0})
FEK513W	ロック・デーモン、登録が失敗しました。 ASID={0}、TCB={1}、USER={2} (Lock daemon, registration failed, ASID={0}, TCB={1}, USER={2})
FEK514W	ロック・デーモン、誤ったログ・レベル、ログ・レベル = {0} (Lock daemon, wrong log level, log level={0})
BPXM023I	(stclock) dataset[(member)] NOT LOCKED
BPXM023I	(stclock) dataset[(member)] LOCKED BY userid
BPXM023I	(stclock) コマンド、誤ったコマンド ((stclock) command, WRONG COMMAND)
BPXM023I	(stclock) コマンド、引数の欠落 ((stclock) command, MISSING ARGUMENT)
BPXM023I	(stclock) 引数、誤った引数 ((stclock) argument, WRONG ARGUMENT)

構文図の読み方

構文図は、入力したコマンドをオペレーティング・システムが正しく解釈できるようにコマンドの指定方法を示したものです。構文図は、左から右、上から下へと水平方向の線（メインパス）をたどって読んでください。

記号

構文図では、以下の記号が使用されます。

記号	説明
>>	構文図の始まりを示します。
>	構文図が続くことを示します。
	構文図のフラグメントか一部分の始まりと終わりを示します。
<<	構文図の終わりを示します。

オペランド

構文図では、以下のタイプのオペランドが使用されます。

- 必須のオペランドは、メインパス行に表示されます。

>>—REQUIRED_OPERAND—<<

- オプションのオペランドは、メインパス行の下に表示されます。

>>
└─OPTIONAL_OPERAND─┘<<

- デフォルト・オペランドは、メインパス行の上に表示されます。

>>
┌─DEFAULT_OPERAND─┐<<

オペランドはキーワードまたは変数として分類されます。

- キーワードは、指定する必要がある定数です。構文図の中でキーワードが大文字と小文字の両方で表記されている場合、大文字の部分はキーワードの省略形です (例: KEYword)。キーワードには大/小文字の区別がありません。大文字か小文字でコーディングできます。
- 変数はイタリック体の英小文字で表記され、ユーザーが指定する名前または値を表します。例えば、データ・セット名は変数です。変数は、大/小文字が区別される場合があります。

構文例

次の例では、USER コマンドはキーワードです。必須の変数パラメーターは `user_id` で、オプションの変数パラメーターは `password` です。これらの変数パラメーターは、ユーザー独自の値に置き換えてください。

```
>>—USER—user_id—┐—————><
                   └password┘
```

非英数字およびブランク・スペース

構文図に英数字以外の文字 (小括弧、ピリオド、コンマ、等号、ブランク・スペースなど) が示されている場合は、その文字を構文の一部としてコーディングする必要があります。この例では、`OPERAND=(001 0.001)` とコーディングする必要があります。

```
>>—OPERAND—==(—001— —0.001—)—————><
```

複数のオペランドの選択

オペランド・グループ内で左に戻る矢印は、複数のオペランドを選択できるか、単一のオペランドを反復できることを意味しています。

```
>>—┐—————><
    └REPEATABLE_OPERAND_1┘
    └REPEATABLE_OPERAND_2┘
    <—┐
```

1 行より長い場合

図が 1 行より長い場合、最初の行は単一の矢印で終わり、2 行目は単一の矢印で始まります。

```
>>—| The first line of a syntax diagram that is longer than one line |—>
>—| The continuation of the subcommands, parameters, or both |—————><
```

構文フラグメント

一部の図には、構文フラグメントが含まれている場合があります。これは、長すぎたり、複雑すぎたり、繰り返しが多すぎる構文を分割するためのものです。構文フラグメント名は大/小文字混合で表記され、構文図の中とフラグメントの見出しに表示されます。フラグメントは、メインの図の下に配置されます。

```
>>—| Syntax fragment |—————><
```

```
Syntax fragment:
|—1ST_OPERAND—,—2ND_OPERAND—,—3RD_OPERAND—|
```

第 9 章 構成問題のトラブルシューティング

この章は、Developer for System z の構成時に起きる可能性があるいくつかの一般的な問題について、ユーザーを支援するためのもので、以下のセクションで構成されています。

- 『FEKLOGS を使用したログとセットアップの分析』
- 146 ページの『ログ・ファイル』
- 152 ページの『ダンプ・ファイル』
- 154 ページの『トレース』
- 157 ページの『z/OS UNIX 許可ビット』
- 161 ページの『予約済み TCP/IP ポート』
- 162 ページの『アドレス・スペース・サイズ』
- 164 ページの『APPC トランザクションおよび TSO コマンド・サービス』
- 165 ページの『各種情報』

詳細は、Developer for System z Web サイト (<http://www-306.ibm.com/software/awdtools/rdz/support/>) の Support セクションで入手できます。このサイトには、弊社のサポート・チームからの最新情報をもたらす技術情報が掲載されています。

この Web サイト (<http://www-306.ibm.com/software/awdtools/rdz/library/>) のライブラリー・セクションには、Developer for System z 資料の最新バージョンが、ホワイト・ペーパーも含めて掲載されています。

Developer for System z インフォメーション・センター (<http://publib.boulder.ibm.com/infocenter/ratdevz/v7r6/index.jsp>) では、Developer for System z クライアント、およびそのクライアントとホストとの対話の方法が (クライアントの視点から) 説明されています。

また、z/OS インターネット・ライブラリー (<http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/>) にも有益な情報があります。

Developer for System z に不足している機能があると思われる場合はお知らせください。機能拡張要求 (RFE) は、次の場所で開示することができます。

<https://www.ibm.com/developerworks/support/rational/rfe/>

FEKLOGS を使用したログとセットアップの分析

Developer for System z には、z/OS UNIX の全ログ・ファイル、および Developer for System z のインストールと構成に関する情報を収集するサンプル・ジョブ FEKLOGS が用意されています。

サンプル・ジョブ FEKLOGS は、FEK.#CUST.JCL に置かれます。ただし、ジョブ FEK.SFEKSAMP (FEKSETUP) をカスタマイズして実行依頼したときに別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

FEKLOGS のカスタマイズは、JCL 内で記述されています。このカスタマイズでは、いくつかの主要な変数を準備します。

注: SDSF をご使用のお客様は、SDSF で **XDC** 行コマンドを使用してジョブ出力をデータ・セットに保存し、それを IBM サポートに提供することもできます。

ログ・ファイル

Developer for System z は、お客様と IBM サポートによる問題の識別と解決に役立つログ・ファイルを作成します。次のリストは、z/OS ホスト・システム上に作成できるログ・ファイルの概要です。これらの製品固有のログとは別に、SYSLOG に関連するメッセージがないかどうか、必ず確認してください。

MVS ベースのログは、該当する DD ステートメントによって見つけることができます。z/OS UNIX ベースのログ・ファイルは、以下のディレクトリに置かれます。

- userlog/\$LOGNAME/

ユーザー固有のログ・ファイルは、userlog/\$LOGNAME に置かれます。ここで、userlog は rsed.envvars 内の user.log ディレクティブと DSTORE_LOG_DIRECTORY ディレクティブを組み合わせた値で、\$LOGNAME はログオン・ユーザー ID (大文字) です。user.log ディレクティブがコメント化されているか存在しない場合は、ユーザーのホーム・パスが使用されます。ホーム・パスはユーザー ID の OMVS セキュリティー・セグメントで定義されます。DSTORE_LOG_DIRECTORY ディレクティブがコメント化されているか存在しない場合は、user.log の値に .eclipse/RSE/ が付加されます。

- .dstoreMemLogging - DataStore メモリ使用率ロギング
- .dstoreTrace - DataStore アクション・ロギング
- fa.log - Fault Analyzer Integration のログ
- fekfivpi.log - fekfivpi IVP テストのログ
- fekfivps.log - fekfivps IVP テストのログ
- ffs.log - ネイティブの MVS 機能を実行する Foreign File System (FFS) サーバーのログ
- ffsget.log - 順次データ・セットまたは PDS メンバーを読み取るファイル読み取りプログラムのログ
- ffsput.log - 順次データ・セットまたは PDS メンバーを書き込むファイル書き込みプログラムのログ
- lock.log - 順次データ・セットまたは PDS メンバーをロック/ロック解除するロック・マネージャーのログ
- rmt_class_loader.cache.jar - RSE リモート・クラス・ローダーによってロードされたクラスのキャッシュ
- rsecomm.log - クライアントからのコマンドを処理する RSE サーバーのログ、および RSE に依存しているすべてのサービスの通信ロギング (Java 例外スタック・トレースを含んでいる場合がある)
- stderr.log - 標準エラー出力 stderr のリダイレクトされたデータ
- stdout.log - 標準出力 stdout のリダイレクトされたデータ

注: .eclipse ディレクトリーおよび .dstore* ログ・ファイルはドット (.) で始まります。これにより、それらは非表示になります。非表示のファイルおよびディレクトリーをリストするには、z/OS UNIX コマンドの **ls -lA** を使用します。Developer for System z クライアントを使用している場合は、「ウィンドウ」>「設定...」>「リモート・システム」>「ファイル」設定ページを選択し、「隠しファイルの表示」を有効にします。

- **daemon-home**

RSE デーモンおよび RSE スレッド・プールに固有のログ・ファイルは、daemon-home に置かれます。ここで、daemon-home は rsed.envvars 内の daemon.log ディレクティブの値です。daemon.log ディレクティブがコメント化されているか存在しない場合は、RSED 開始タスクに割り当てられているユーザー ID のホーム・ディレクトリーが使用されます。ホーム・ディレクトリーはユーザー ID の OMVS セキュリティー・セグメントで定義されます。

- rsedaemon.log - RSE デーモンのログ
- rseserver.log - RSE スレッド・プールのログ
- audit.log - RSE 監査証跡
- serverlogs.count - RSE スレッド・プール・ストリームをログに記録するためのカウンター
- stderr.*.log - RSE スレッド・プールの標準エラー・ストリーム
- stdout.*.log - RSE スレッド・プールの標準出力ストリーム

注: オペレーター・コマンドを使用して、上記のいくつかのログ・ファイルにデータを書き込まれる量を制御できます。詳細については、133 ページの『第 8 章 オペレーター・コマンド』を参照してください。

JES ジョブ・モニター・ロギング

- **SYSOUT DD**

通常操作のロギング。サンプル JCL FEK.#CUST.PROCLIB(JMON) 内のデフォルト値は SYSOUT=* です。

- **SYSPRINT DD**

トレース・ロギング。サンプル JCL FEK.#CUST.PROCLIB(JMON) 内のデフォルト値は SYSOUT=* です。トレースは -TV パラメーターでアクティブにされます。詳細については、154 ページの『JES ジョブ・モニターのトレース』を参照してください。

ロック・デーモン・ロギング

- **STDOUT DD**

Java 標準出力 stdout のリダイレクトされたデータ。サンプル JCL FEK.#CUST.PROCLIB(LOCKD) 内のデフォルト値は SYSOUT=* です。

- **STDERR DD**

Java 標準エラー出力 stderr のリダイレクトされたデータ。サンプル JCL FEK.#CUST.PROCLIB(LOCKD) 内のデフォルト値は SYSOUT=* です。

RSE デーモンおよびスレッド・プールのロギング

- **STDOUT DD**

RSE デーモンの Java 標準出力 `stdout` のリダイレクトされたデータ。サンプル JCL `FEK.#CUST.PROCLIB(RSED)` 内のデフォルト値は `SYSOUT=*` です。

- **STDERR DD**

RSE デーモンの Java 標準エラー出力 `stderr` のリダイレクトされたデータ。サンプル JCL `FEK.#CUST.PROCLIB(RSED)` 内のデフォルト値は `SYSOUT=*` です。

- **daemon-home**

RSE デーモンおよび RSE スレッド・プールに固有のログ・ファイルは、`daemon-home` に置かれます。ここで、`daemon-home` は `rsed.envvars` 内の `daemon.log` ディレクティブの値です。`daemon.log` ディレクティブがコメント化されているか存在しない場合は、RSED 開始タスクに割り当てられているユーザー ID のホーム・ディレクトリーが使用されます。ホーム・ディレクトリーはユーザー ID の OMVS セキュリティー・セグメントで定義されます。

- `rsedaemon.log` - RSE デーモンのログ
- `rseserver.log` - RSE スレッド・プールのログ
- `audit.log` - RSE 監査証跡
- `serverlogs.count` - RSE スレッド・プール・ストリームをログに記録するためのカウンター
- `stderr.*.log` - RSE スレッド・プールの標準エラー・ストリーム
- `stdout.*.log` - RSE スレッド・プールの標準出力ストリーム

注:

- `serverlogs.count`、`stderr.*.log`、および `stdout.*.log` は、`rsed.envvars` 内の `enable.standard.log` ディレクティブがアクティブである場合、またはこの機能が **modify rsestandardlog on** オペレーター・コマンドで動的にアクティブ化される場合にのみ作成されます。
- `stderr.*.log` および `stdout.*.log` 内の `*` は、デフォルトでは 1 です。ただし、複数の RSE スレッド・プールが存在することが可能であるため、ファイル名が固有となるように、新しい RSE スレッド・プールが増えるたびにこの数値に 1 が加算されます。
- `enable.standard.log` ディレクティブがアクティブであるときは、ユーザー固有の `stdout.log` および `stderr.log` ログ・ファイルが存在しません。ユーザー固有のデータは、対応する RSE スレッド・プール・ストリームに書き込まれます。
- オペレーター・コマンドを使用して、上記のいくつかのログ・ファイルにデータを書き込まれる量を制御できます。詳細については、133 ページの『第 8 章 オペレーター・コマンド』を参照してください。

RSE ユーザー・ロギング

- **userlog/\$LOGNAME/**

RSE に関連するコンポーネントによって作成される複数のログ・ファイルがあります。これらはいずれも `userlog/$LOGNAME` に置かれます。ここで、`userlog` は `rsed.envvars` 内の `user.log` ディレクティブと `DSTORE_LOG_DIRECTORY` ディレクティブを組み合わせた値で、`$LOGNAME` はログオン・ユーザー ID (大文字) です。`user.log` ディレクティブがコメント化されているか存在しない場合は、ユーザーのホーム・パスが使用されます。ホーム・パスはユーザー ID の OMVS セキュリティー・セグメントで定義されます。`DSTORE_LOG_DIRECTORY` ディレクティブがコメント化されているか存在しない場合は、`user.log` の値に `.eclipse/RSE/` が付加されます。

- `.dstoreMemLogging` - DataStore メモリー使用率ロギング
- `.dstoreTrace` - DataStore アクション・ロギング
- `ffs.log` - ネイティブの MVS 機能を実行する Foreign File System (FFS) サーバーのログ
- `ffsget.log` - 順次データ・セットまたは PDS メンバーを読み取るファイル読み取りプログラムのログ
- `ffsput.log` - 順次データ・セットまたは PDS メンバーを書き込むファイル書き込みプログラムのログ
- `lock.log` - 順次データ・セットまたは PDS メンバーをロックまたはロック解除するロック・マネージャーのログ
- `rmt_class_loader.cache.jar` - RSE リモート・クラス・ローダーによってロードされたクラスのキャッシュ
- `rsecomm.log` - クライアントからのコマンドを処理する RSE サーバーのログ、および RSE に依存しているすべてのサービスの通信ロギング (Java 例外スタック・トレースを含んでいる場合がある)
- `stderr.log` - 標準エラー出力 `stderr` のリダイレクトされたデータ
- `stdout.log` - 標準出力 `stdout` のリダイレクトされたデータ

注:

- `.eclipse` ディレクトリーおよび `.dstore*` ログ・ファイルはドット (.) で始まります。これにより、それらは非表示になります。非表示のファイルおよびディレクトリーをリストするには、`z/OS UNIX` コマンドの `ls -lA` を使用します。Developer for System z クライアントを使用している場合は、「ウィンドウ」>「設定...」>「リモート・システム」>「ファイル」設定ページを選択し、「隠しファイルの表示」を有効にします。
- `.dstore*` ログ・ファイルの作成は、46 ページの『`_RSE_JAVAOPTS` での追加 Java 始動パラメーターの定義』の説明にあるように、`-DDSTORE_*` Java 始動オプションによって制御されます。
- `.dstore*` ログ・ファイルは、ASCII で作成されます。これらのログ・ファイルを EBCDIC で表示する (コード・ページ IBM-1047 を使用している場合) には、`z/OS UNIX` コマンドの `iconv -f ISO8859-1 -t IBM-1047 .dstore*` を使用します。
- `enable.standard.log` ディレクティブがアクティブであるときは、ユーザー固有の `stdout.log` および `stderr.log` ログ・ファイルが存在しません。ユーザー固有のデータは、対応する RSE スレッド・プール・ストリームに書き込まれます。

- オペレーター・コマンドを使用して、上記のいくつかのログ・ファイルにデータを書き込まれる量を制御できます。詳細については、133 ページの『第 8 章 オペレーター・コマンド』を参照してください。

Fault Analyzer Integration ロギング

- `userlog/$LOGNAME/`

Fault Analyzer Integration ロギング。ここで、`userlog` は `rsed.envvars` 内の `user.log` ディレクティブと `DSTORE_LOG_DIRECTORY` ディレクティブを組み合わせた値で、`$LOGNAME` はログオン・ユーザー ID (大文字) です。`user.log` ディレクティブがコメント化されているか存在しない場合は、ユーザーのホーム・パスが使用されます。ホーム・パスはユーザー ID の OMVS セキュリティー・セグメントで定義されます。`DSTORE_LOG_DIRECTORY` ディレクティブがコメント化されているか存在しない場合は、`user.log` の値に `.eclipse/RSE/` が付加されます。

- `fa.log` - Fault Analyzer Integration のログ
- `rsecomm.log` - Fault Analyzer Integration の通信ロギング

File Manager Integration ロギング

- `userlog/$LOGNAME/rsecomm.log`

File Manager Integration の通信ロギング。ここで、`userlog` は `rsed.envvars` 内の `user.log` ディレクティブと `DSTORE_LOG_DIRECTORY` ディレクティブを組み合わせた値で、`$LOGNAME` はログオン・ユーザー ID (大文字) です。`user.log` ディレクティブがコメント化されているか存在しない場合は、ユーザーのホーム・パスが使用されます。ホーム・パスはユーザー ID の OMVS セキュリティー・セグメントで定義されます。`DSTORE_LOG_DIRECTORY` ディレクティブがコメント化されているか存在しない場合は、`user.log` の値に `.eclipse/RSE/` が付加されます。

SCLM Developer Toolkit のロギング

- `userlog/$LOGNAME/rsecomm.log`

SCLM Developer Toolkit の通信ロギング。ここで、`userlog` は `rsed.envvars` 内の `user.log` ディレクティブと `DSTORE_LOG_DIRECTORY` ディレクティブを組み合わせた値で、`$LOGNAME` はログオン・ユーザー ID (大文字) です。`user.log` ディレクティブがコメント化されているか存在しない場合は、ユーザーのホーム・パスが使用されます。ホーム・パスはユーザー ID の OMVS セキュリティー・セグメントで定義されます。`DSTORE_LOG_DIRECTORY` ディレクティブがコメント化されているか存在しない場合は、`user.log` の値に `.eclipse/RSE/` が付加されます。

CARMA ロギング

- CARMA サーバー・ジョブ

バッチ・インターフェースを使用して CARMA との接続を開くと、FEK.#CUST.SYSPROC(CRASUBMT) は CRAport というサーバー・ジョブを (ユーザーの、所有者としてのユーザー ID を使用して) 開始します。ここで、port は使用される TCP/IP ポートです。

- **CARMALOG DD**

選択された CARMA 始動方式で DD ステートメント CARMALOG が指定されている場合、CARMA ロギングはサーバー・ジョブでその DD ステートメントへリダイレクトされ、指定されていない場合は SYSPRINT へ送られます。

- **SYSPRINT DD**

サーバー・ジョブの SYSPRINT は、DD ステートメント CARMALOG が定義されていない場合、CARMA ロギングを保持します。

- **userlog/\$LOGNAME/rsecomm.log**

CARMA の通信ロギング。ここで、userlog は rsed.envvars 内の user.log ディレクティブと DSTORE_LOG_DIRECTORY ディレクティブを組み合わせた値で、\$LOGNAME はログオン・ユーザー ID (大文字) です。user.log ディレクティブがコメント化されているか存在しない場合は、ユーザーのホーム・パスが使用されます。ホーム・パスはユーザー ID の OMVS セキュリティー・セグメントで定義されます。DSTORE_LOG_DIRECTORY ディレクティブがコメント化されているか存在しない場合は、user.log の値に .eclipse/RSE/ が付加されます。

APPC トランザクション (TSO コマンド・サービス) ロギング

- **SYSPRINT DD**

APPC 管理ユーティリティーは、トランザクション・プログラム (TP) プロファイルを追加および変更するとき、TP プロファイルとその JCL に構文エラーがないかどうか検査します。このフェーズからの出力は、TP プロファイル構文エラー・メッセージ、ユーティリティー処理メッセージ、および JCL 変換ステートメントからなっています。このフェーズからのメッセージのロギングは、ATBSDFMU ユーティリティーの SYSPRINT DD ステートメントによって制御されます。サンプル JCL FEK.SFEKSAMP(FEKAPPCC) 内のデフォルト値は SYSOUT=* です。詳細については、「MVS 計画: APPC/MVS 管理」(SA88-8571) を参照してください。

- **&SYSUID.FEKFRSRV.&TPDATE.&TPTIME.LOG**

TP が実行された場合、TP ランタイム・メッセージ (割り振りメッセージや終了メッセージなど) は TP プロファイル内の MESSAGE_DATA_SET キーワードで指定されたログへ送られます。サンプル JCL FEK.SFEKSAMP(FEKAPPCC) 内のデフォルト値は &SYSUID.FEKFRSRV.&TPDATE.&TPTIME.LOG です。詳細については、「MVS 計画: APPC/MVS 管理」(SA88-8571) を参照してください。

注: APPC 定義とサイトのデフォルトによっては、トランザクション定義に KEEP_MESSAGE_LOG(ALWAYS) キーワードを追加しないと、このログ・ファイルが表示されない場合があります。これについての詳細は、「MVS 計画: APPC/MVS 管理」(SA88-8571) を参照してください。

fekfivpi IVP テスト・ロギング

- `userlog/$LOGNAME/fekfivpi.log`

fekfivpi -file コマンド (TSO/ISPF クライアント・ゲートウェイに関連した IVP テスト) の出力。ここで、userlog は rsed.envvars 内の user.log ディレクティブと DSTORE_LOG_DIRECTORY ディレクティブを組み合わせた値で、\$LOGNAME はログオン・ユーザー ID (大文字) です。user.log ディレクティブがコメント化されているか存在しない場合は、ユーザーのホーム・パスが使用されます。ホーム・パスはユーザー ID の OMVS セキュリティー・セグメントで定義されます。DSTORE_LOG_DIRECTORY ディレクティブがコメント化されているか存在しない場合は、user.log の値に .eclipse/RSE/ が付加されます。

fekfivps IVP テスト・ロギング

- `userlog/$LOGNAME/fekfivps.log`

fekfivps -file コマンド (SCLMDT に関連した IVP テスト) の出力。ここで、userlog は rsed.envvars 内の user.log ディレクティブと DSTORE_LOG_DIRECTORY ディレクティブを組み合わせた値で、\$LOGNAME はログオン・ユーザー ID (大文字) です。user.log ディレクティブがコメント化されているか存在しない場合は、ユーザーのホーム・パスが使用されます。ホーム・パスはユーザー ID の OMVS セキュリティー・セグメントで定義されます。DSTORE_LOG_DIRECTORY ディレクティブがコメント化されているか存在しない場合は、user.log の値に .eclipse/RSE/ が付加されます。

ダンプ・ファイル

製品が異常終了した場合、問題判別を支援するためにストレージ・ダンプが作成されます。これらのダンプの可用性とロケーションは、サイト固有の設定に大きく依存します。したがって、ダンプが作成されなかったり、以下の説明と異なるロケーションに作成されたりする場合があります。

MVS ダンプ

プログラムを MVS 内で実行している場合は、システム・ダンプ・ファイルを検査し、JCL に以下の DD ステートメント (製品によって異なります) があるかどうかを確認してください。

- SYSABEND
- SYSMDUMP
- SYSUDUMP
- CEEDUMP
- SYSPRINT
- SYSOUT

これらの DD ステートメントの詳細については、「*MVS JCL 解説書*」(SA88-8569) および「*Language Environment デバッグ・ガイド*」(GA88-8548) を参照してください。

Java ダンプ

z/OS UNIX では、大部分の Developer for System z ダンプが Java 仮想マシン (JVM) によって制御されます。

JVM は、デフォルトでは初期化時にダンプ・エージェント・セット (SYSTDUMP と JAVADUMP) を作成します。このダンプ・エージェント・セットは、`JAVA_DUMP_OPTS` 環境変数を使用してオーバーライドでき、さらに、コマンド行で `-Xdump` を使用することによってオーバーライドできます。JVM コマンド行オプションは、`rsed.envvars` の `_RSE_JAVA_OPTS` ディレクティブで定義されます。IBM サポートから指示された場合以外は、ダンプの設定を一切変更しないでください。

注: コマンド行の `-Xdump:what` オプションを使用すると、始動完了時にどのダンプ・エージェントが存在するかを決定できます。

生成できるダンプのタイプは、以下のとおりです。

SYSTDUMP

Java トランザクション・ダンプ。z/OS によって生成される不定形式のストレージ・ダンプ。

ダンプは、`%uid.JVM.TDUMP.%job.D%y%m%d.T%H%M%S` という書式のデフォルト名、または `JAVA_DUMP_TDUMP_PATTERN` 環境変数の設定によって決まるデフォルト名を使用して、順次 MVS データ・セットに書き込まれます。トランザクション・ダンプを作成したくない場合は、環境変数 `IBM_JAVA_ZOS_TDUMP=NO` を `rsed.envvars` に追加します。

注: `JAVA_DUMP_TDUMP_PATTERN` を使用すると変数を使用できます。それらの変数は、トランザクション・ダンプの取得時に実際の値に変換されます。

表 20. `JAVA_DUMP_TDUMP_PATTERN` 変数

変数	使用法
<code>%uid</code>	ユーザー ID
<code>%job</code>	ジョブ名
<code>%y</code>	年 (2 桁)
<code>%m</code>	月 (2 桁)
<code>%d</code>	日 (2 桁)
<code>%H</code>	時間 (2 桁)
<code>%M</code>	分 (2 桁)
<code>%S</code>	秒 (2 桁)

CEEDUMP

言語環境プログラム (LE) ダンプ。定様式の要約システム・ダンプ。これは JVM プロセス内の各スレッドのスタック・トレースを、レジスター情報や各レジスターのストレージの短いダンプと一緒に表示します。

ダンプは `CEEDUMP.yyyymmdd.hhmmss.pid` という名前の z/OS UNIX ファイルに書き込まれます。ここで、`yyymmdd` は現在の日付で、`hhmmss` は現在の

時刻、pid は現行プロセス ID です。このファイルの可能なロケーションについては、『z/OS UNIX ダンプ・ロケーション』に説明があります。

HEAPDUMP

Java ヒープ上にあるオブジェクトの定様式ダンプ (リスト)。

ダンプは HEAPDUMP.yyyyymmdd.hhmmss.pid.TXT という名前の z/OS UNIX ファイルに書き込まれます。ここで、yyyymmdd は現在の日付で、hhmmss は現在の時刻、pid は現行プロセス ID です。このファイルの可能なロケーションについては、『z/OS UNIX ダンプ・ロケーション』に説明があります。

JAVADUMP

JVM の定様式分析。これには JVM と Java アプリケーションに関連した診断情報 (例えば、アプリケーション環境、スレッド、ネイティブ・スタック、ロック、メモリーなど) が入っています。

ダンプは JAVADUMP.yyyyymmdd.hhmmss.pid.TXT という名前の z/OS UNIX ファイルに書き込まれます。ここで、yyyymmdd は現在の日付で、hhmmss は現在の時刻、pid は現行プロセス ID です。このファイルの可能なロケーションについては、『z/OS UNIX ダンプ・ロケーション』に説明があります。

JVM ダンプの詳細については「*Java Diagnostic Guide*」(SC34-6358) を、また、LE 固有の情報については「*Language Environment デバッグ・ガイド*」(GA88-8548) を参照してください。

z/OS UNIX ダンプ・ロケーション

JVM は、以下の各ロケーションの有無と書き込み権限を検査し、最初に使用可能なロケーションに CEEDUMP、HEAPDUMP、および JAVADUMP ファイルを保管します。ダンプ・ファイルを正しく書き込むための十分なディスク・スペースが必要であることに注意してください。

1. 環境変数 _CEE_DMPTARG 内のディレクトリー (ある場合)。この変数は、rsed.envvars 内で /tmp として設定されます。これを /dev/null に変更して、ダンプ・ファイルが作成されないようにすることができます。
2. 現行作業ディレクトリー (このディレクトリーがルート・ディレクトリー (/) でなく、書き込み可能である場合)。
3. 環境変数 TMPDIR (一時ディレクトリーが /tmp でない場合に、そのロケーションを示す環境変数) 内のディレクトリー(ある場合)。
4. /tmp ディレクトリー。
5. ダンプは、上記のどこにも保管できない場合、stderr へ書き込まれます。

トレース

JES ジョブ・モニターのトレース

JES ジョブ・モニターのトレースは、133 ページの『第 8 章 オペレーター・コマンド』の説明にあるように、システム・オペレーターによって制御されます。

- PRM=-TV パラメーターを指定して JMON 開始タスクを始動すると、冗長モード (トレース) がアクティブになります。
- **modify -TV** コマンドおよび **modify -TN** コマンドはトレースのアクティブ化および非アクティブ化を行います。

RSE トレース

RSE に関連するコンポーネントによって作成される複数のログ・ファイルがあります。そのほとんどは `userlog/$LOGNAME` に置かれます。ここで、`userlog` は `rsed.envvars` 内の `user.log` ディレクティブと `DSTORE_LOG_DIRECTORY` ディレクティブを組み合わせた値で、`$LOGNAME` はログオン・ユーザー ID (大文字) です。`user.log` ディレクティブがコメント化されているか存在しない場合は、ユーザーのホーム・パスが使用されます。ホーム・パスはユーザー ID の OMVS セキュリティー・セグメントで定義されます。`DSTORE_LOG_DIRECTORY` ディレクティブがコメント化されているか存在しない場合は、`user.log` の値に `.eclipse/RSE/` が付加されます。

`ffs*.log`、`lock.log`、および `rsecomm.log` に書き込まれるデータの量は、**modify rsecommlog** オペレーター・コマンドによって制御するか、`rsecomm.properties` 内で `debug_level` を設定することによって制御します。詳細については、133 ページの『第 8 章 オペレーター・コマンド』、および 102 ページの『(オプション) RSE トレース』を参照してください。

`.dstore*` ログ・ファイルの作成は、46 ページの『`_RSE_JAVAOPTS` での追加 Java 始動パラメーターの定義』の説明にあるように、`-DDSTORE_*` Java 始動オプションによって制御されます。

注:

- `.eclipse` ディレクトリーおよび `.dstore*` ログ・ファイルはドット (.) で始まります。これにより、それらは非表示になります。非表示のファイルおよびディレクトリーをリストするには、`z/OS UNIX` コマンドの **ls -lA** を使用します。Developer for System z クライアントを使用している場合は、「ウィンドウ」>「設定...」>「リモート・システム」>「ファイル」設定ページを選択し、「隠しファイルの表示」を有効にします。
- `.dstore*` ログ・ファイルは、ASCII で作成されます。これらのログ・ファイルを EBCDIC で表示する (コード・ページ IBM-1047 を使用している場合) には、`z/OS UNIX` コマンドの **iconv -f ISO8859-1 -t IBM-1047 .dstore*** を使用します。

RSE デーモンおよび RSE スレッド・プールに固有のログ・ファイルは、`daemon-home` に置かれます。ここで、`daemon-home` は `rsed.envvars` 内の `daemon.log` ディレクティブの値です。`daemon.log` ディレクティブがコメント化されているか存在しない場合は、RSED 開始タスクに割り当てられているユーザー ID のホーム・ディレクトリーが使用されます。ホーム・ディレクトリーはユーザー ID の OMVS セキュリティー・セグメントで定義されます。

`rsedaemon.log` および `rserver.log` に書き込まれるデータの量は、**modify rsedaemonlog** および **modify rserverlog** オペレーター・コマンドによって制御するか、`rsecomm.properties` で `debug_level` を設定することによって制御します。

詳細については、133 ページの『第 8 章 オペレーター・コマンド』、および 102 ページの『(オプション) RSE トレース』を参照してください。

serverlogs.count、stderr.*.log、および stdout.*.log は、rsed.envvars 内の enable.standard.log ディレクティブがアクティブである場合、またはこの機能が **modify rsestandardlog on** オペレーター・コマンドで動的にアクティブ化される場合にのみ作成されます。

ロック・デーモンのトレース

ロック・デーモン固有のログは、LOCKD 開始タスクの STDOUT DD に置かれます。ログに書き込まれるデータの量は、LOG 始動パラメーターによって制御されます。詳細については、133 ページの『第 8 章 オペレーター・コマンド』、および 102 ページの『(オプション) RSE トレース』を参照してください。

CARMA トレース

ユーザーは、クライアント上の CARMA 接続のプロパティ・タブで「トレース・レベル」を設定することにより、CARMA が生成するトレース情報の量を制御できます。「トレース・レベル」の選択項目は、以下のとおりです。

- ログングを使用不可に設定
- エラー・ログング
- 警告ログング
- 通知ログング
- デバッグ・ログング

デフォルト値は以下のとおりです。

エラー・ログング

ログ・ファイルのロケーションの詳細については、146 ページの『ログ・ファイル』を参照してください。

エラー・フィードバック・トレース

以下のプロシージャーを使用すると、リモート・ビルド・プロシージャーでのエラー・フィードバック問題を診断するために必要な情報を収集できます。このトレースはパフォーマンス低下の原因になるため、IBM サポートの指示の下でのみ実行してください。このセクションで、hlq と表記したものはすべて、Developer for System z のインストール時に使用した高位修飾子を指しています。インストールのデフォルトは FEK ですが、ご使用のサイトには当てはまらない場合があります。

1. アクティブな ELAXFCOC コンパイル・プロシージャーのバックアップ・コピーを作成してください。このプロシージャーは、デフォルトでは出荷時にデータ・セット hlq.SFEKSAMP に組み込まれていますが、27 ページの『ELAXF* リモート・ビルド・プロシージャー』で説明されているように、別のロケーション (SYS1.PROCLIB など) にコピーされている可能性があります。
2. アクティブな ELAXFCOC プロシージャーを変更して、EXIT(ADEXIT(ELAXMGUX)) コンパイル・オプションに「MAXTRACE」ストリングを組み込みます。

```
//COBOL EXEC PGM=IGYCRCTL,REGION=2048K,  
//*          PARM=('EXIT(ADEXIT(ELAXMGUX))',  
//          PARM=('EXIT(ADEXIT('MAXTRACE',ELAXMGUX))',
```



```
//          'ADATA',
//          'LIB',
//          'TEST(NONE,SYM,SEP)',
//          'LIST',
//          'FLAG(I,I)'&CICS &DB2 &COMP)
```

注: MAXTRACE を二重のアポストロフィで囲む必要があります。このオプションは、次のようになります。EXIT(ADEXIT('MAXTRACE',ELAXMGUX))。

3. 詳細にトレースしたい COBOL プログラムに対し、リモート構文検査を実行します。
4. JES 出力の SYSOUT 部分は、SIDEFILE1、SIDEFILE2、SIDEFILE3、および SIDEFILE4 のデータ・セット名のリストで始まります。

```
ABOUT TOO OPEN SIDEFILE1 - NAME = 'uid.DT021207.TT110823.M0000045.C0000000'
SUCCESSFUL OPEN SIDEFILE1 - NAME = 'uid.DT021207.TT110823.M0000045.C0000000'
ABOUT TOO OPEN SIDEFILE2 - NAME = 'uid.DT021207.TT110823.M0000111.C0000001'
SUCCESSFUL OPEN SIDEFILE2 - NAME = 'uid.DT021207.TT110823.M0000111.C0000001'
ABOUT TOO OPEN SIDEFILE3 - NAME = 'uid.DT021207.TT110823.M0000174.C0000002'
SUCCESSFUL OPEN SIDEFILE3 - NAME = 'uid.DT021207.TT110823.M0000174.C0000002'
ABOUT TOO OPEN SIDEFILE4 - NAME = 'uid.DT021207.TT110823.M0000236.C0000003'
SUCCESSFUL OPEN SIDEFILE4 - NAME = 'uid.DT021207.TT110823.M0000236.C0000003'
```

注: 設定によっては、SIDEFILE1 および SIDEFILE2 が DD ステートメントを指している場合があります (SUCCESSFUL OPEN SIDEFILE1 - NAME = DD:WSEDSF1)。実際のデータ・セット名を得るには、出力の JESJCL 部分 (これは SYSOUT 部分の前に置かれます) を参照してください。

```
22 //COBOL.WSEDSF1 DD DISP=MOD,
// DSN=uid.ERRCOB.member.SF.Z682746.XML
23 //COBOL.WSEDSF2 DD DISP=MOD,
// DSN=uid.ERRCOB.member.SF.Z682747.XML
```

5. これら 4 つのデータ・セットを PC にコピーします。例えば、Developer for System z 内にローカル COBOL プロジェクトを作成し、SIDEFILE1->4 データ・セットを追加します。
6. 完全な JES ジョブ・ログを PC にコピーします。例えば、Developer for System z でジョブ出力を開き、それをローカル・プロジェクトに保存するために「ファイル」>「別名保存...」を選択します。
7. プロシージャ ELAXFCOC を元の状態に復元します。これは、変更を元に戻す (コンパイル・オプション内の「MAXTRACE」ストリングを除去する) か、バックアップをリストアします。
8. 収集したファイル (SIDEFILE1->4 およびジョブ・ログ) を IBM サポートへ送ります。

z/OS UNIX 許可ビット

Developer for System z では、z/OS UNIX ファイル・システムおよび一部の z/OS UNIX ファイルに特定の許可ビットがセットされている必要があります。

SETUID ファイル・システム属性

リモート・システム・エクスプローラー (RSE) は、クライアントをホストに接続するなどのコア・サービスを提供する Developer for System z コンポーネントです。これは、ユーザーのセキュリティ環境を作成するなどのタスクの実行を許可されている必要があります。

Developer for System z がインストールされているファイル・システム (HFS または zFS) は、SETUID 許可ビットをオン (これはデフォルトです) にしてマウントする必要があります。NOSETUID パラメーターを指定してファイル・システムをマウントすると、ユーザーのセキュリティ環境が Developer for System z によって作成されず、接続要求が失敗します。

TSO の **ISHELL** コマンドを使用して、SETUID ビットの現行ステータスをリストします。ISHELL パネルで、「**File_systems**」>「**1. マウント・テーブル... (1. Mount table...)**」を選択して、マウントされたファイル・システムをリストします。**a** 行コマンドは、選択されたファイル・システムの属性を表示します。ここで、「Ignore SETUID」フィールドは 0 である必要があります。

プログラム制御許可

リモート・システム・エクスプローラー (RSE) は、クライアントをホストに接続するなどのコア・サービスを提供する Developer for System z コンポーネントです。これは、クライアントのユーザー ID への切り替えなどのタスクを実行するために、プログラム制御で実行する必要があります。

z/OS UNIX プログラム制御ビットは、SMP/E インストールのときに、必要ならば設定されます。ただし、169 ページの『第 10 章 セキュリティーに関する考慮事項』の説明にあるように、セキュリティ製品への Java インターフェースは除きます。この許可ビットは、Developer for System z ディレクトリーの手動コピー中に保存しなかった場合、失われることがあります。

以下の Developer for System z ファイルはプログラムで制御される必要があります。

- /usr/lpp/rdz/bin/
 - fekfdivp
 - fekfomvs
 - fekfrivp
- /usr/lpp/rdz/lib/
 - fekfdir.dll
 - libfekdcore.so
 - libfekfmain.so
- /usr/lpp/rdz/lib/icuc/
 - libicudata.dll
 - libicudata40.1.dll
 - libicudata40.dll
 - libicudata64.40.1.dll
 - libicudata64.40.dll
 - libicudata64.dll
 - libicuuc.dll
 - libicuuc40.1.dll
 - libicuuc40.dll

```
|
|      - libicuuc64.40.1.dll
|
|      - libicuuc64.40.dll
|
|      - libicuuc64.dll
```

| 注: libicu*64.* ファイルは、APAR AM07305 に対応して 64 ビットのサポートを
| 有効にする Developer for System z PTF を適用した場合にのみ存在します。

z/OS UNIX コマンド **ls -E** を使用して、拡張属性をリストします。このリストで、
プログラム制御ビットには、次のサンプルに示すように英字の p のマークが付いま
す (\$ は z/OS UNIX プロンプトです)。

```
$ cd /usr/lpp/rdz
$ ls -E lib/fekf*
-rwxr-xr-x -ps- 2 user      group      94208 Jul  8 12:31 lib/fekfdir.dll
```

プログラム制御ビットを手動でセットするには、次のサンプルに示すように、 z/OS
UNIX コマンド **extattr +p** を使用します (\$ および # は z/OS UNIX プロンプト
です)。

```
$ cd /usr/lpp/rdz
$ su
# extattr +p lib/fekf*
# exit
$ ls -E lib/fekf*
-rwxr-xr-x -ps- 2 user      group      94208 Jul  8 12:31 lib/fekfdir.dll
```

注: **extattr +p** コマンドを使用するには、少なくとも、セキュリティ・ソフトウ
ェアの FACILITY クラス内の BPX.FILEATTR.PROGCTL プロファイルに対する
READ アクセス権が必要です。あるいは、このプロファイルが定義されていな
い場合は、スーパーユーザー (UID 0) であることが必要です。詳細について
は、「UNIX System Services 計画」(GA88-8639) を参照してください。

| APF 許可

| リモート・システム・エクスプローラー (RSE) は、クライアントをホストに接続す
| るなどのコア・サービスを提供する Developer for System z のコンポーネントで
| す。詳細なプロセス・リソース使用量を表示するなどのタスクを実行するため
| は、このコンポーネントを APF 許可がある状態で実行する必要があります。

| z/OS UNIX APF ビットは、SMP/E インストールのときに、必要ならば設定されま
| す。この許可ビットは、Developer for System z ディレクトリーの手動コピー中に保
| 存しなかった場合、失われることがあります。

| 以下の Developer for System z ファイルには、APF 許可があることが必要です。

```
| • /usr/lpp/rdz/bin/
|
|      - fekfomvs
|
|      - fekfrivp
```

| 拡張属性をリストするには、z/OS UNIX コマンド **ls -E** を使用します。このリスト
| では、次のサンプルに示すように、APF ビットに英字の a のマークが付きます (\$
| は z/OS UNIX プロンプトです)。

```
| $ cd /usr/lpp/rdz
| $ ls -E bin/fekfrivp
|-rwxr-xr-x aps- 2 user      group      114688 Sep 17 06:41 bin/fekfrivp
```

APF ビットを手動でセットするには、次のサンプルに示すように、z/OS UNIX コマンド **extattr +a** を使用します (\$ および # は z/OS UNIX プロンプトです)。

```
$ cd /usr/lpp/rdz
$ su
# extattr +a bin/fekfrivp
# exit
$ ls -E bin/fekfrivp
-rwxr-xr-x  aps-  2 user      group      114688 Sep 17 06:41 bin/fekfrivp
```

注: **extattr +a** コマンドを使用するには、少なくとも、セキュリティー・ソフトウェアの FACILITY クラス内の BPX.FILEATTR.APF プロファイルに対する READ アクセス権が必要です。あるいは、このプロファイルが定義されていない場合は、スーパーユーザー (UID 0) であることが必要です。詳細については、「UNIX System Services 計画」(GA88-8639) を参照してください。

スティッキー・ビット

オプションの Developer for System z サービスの中には、MVS ロード・モジュールを z/OS UNIX で使用可能にしなければならないものもあります。これを行うには、z/OS UNIX で、スティッキー・ビットをオンにしたスタブ (ダミー・ファイル) を作成します。スタブが実行されると、z/OS UNIX は同じ名前の MVS ロード・モジュールを探し、代わりにそのロード・モジュールを実行します。

z/OS UNIX スティッキー・ビットは、SMP/E インストールのときに、必要ならば設定されます。これらの許可ビットは、Developer for System z の手動コピーのときに保存されなかった場合は、失われることがあります。

以下の Developer for System z ファイルは、スティッキー・ビットがオンであることが必要です。

- /usr/lpp/rdz/bin/
 - BWBTSOW
 - CRASTART

z/OS UNIX コマンド **ls -l** を使用して、許可をリストします。このリストで、スティッキー・ビットには、次のサンプルに示すように英字の **t** のマークが付きます (\$ は z/OS UNIX プロンプトです)。

```
$ cd /usr/lpp/rdz
$ ls -l bin/CRA*
-rwxr-xr-t  2 user      group      71 Jul  8 12:31 bin/CRASTART
```

スティッキー・ビットを手動でセットするには、次のサンプルに示すように、z/OS UNIX コマンド **chmod +t** を使用します (\$ および # は z/OS UNIX プロンプトです)。

```
$ cd /usr/lpp/rdz
$ su
# chmod +t bin/CRA*
# exit
$ ls -l bin/CRA*
-rwxr-xr-t  2 user      group      71 Jul  8 12:31 bin/CRASTART
```

注: **chmod** コマンドを使用するには、少なくとも、セキュリティー・ソフトウェアの UNIXPRIV クラス内の SUPERUSER.FILESYS.CHANGEPERMS プロファイルに対する READ アクセス権が必要です。あるいは、このプロファイルが定義されてい

ない場合は、スーパーユーザー (UID 0) であることが必要です。詳細については、「UNIX System Services 計画」(GA88-8639) を参照してください。

予約済み TCP/IP ポート

netstat コマンド (TSO または z/OS UNIX) を使用すると、現在使用中のポートの概要を取得できます。このコマンドの出力は、下記の例のようになります。使用されているポートは、「Local Socket」列の最後の番号 (「..」の後) です。これらのポートは既に使用されているので、Developer for System z 構成に使用することはできません。

IPv4

MVS TCP/IP NETSTAT CS VxRy		TCPIP Name: TCPIP		16:36:42
User Id	Conn	Local Socket	Foreign Socket	State
-----	----	-----	-----	----
BPXOINIT	00000014	0.0.0.0..10007	0.0.0.0..0	Listen
INETD4	0000004D	0.0.0.0..512	0.0.0.0..0	Listen
RSED	0000004B	0.0.0.0..4035	0.0.0.0..0	Listen
JMON	00000038	0.0.0.0..6715	0.0.0.0..0	Listen

IPv6

MVS TCP/IP NETSTAT CS VxRy		TCPIP Name: TCPIP		12:46:25
User Id	Conn	State		
-----	----	----		
BPXOINIT	00000018	Listen		
	Local Socket:	0.0.0.0..10007		
	Foreign Socket:	0.0.0.0..0		
INETD4	00000046	Listen		
	Local Socket:	0.0.0.0..512		
	Foreign Socket:	0.0.0.0..0		
RSED	0000004B	Listen		
	Local Socket:	0.0.0.0..4035		
	Foreign Socket:	0.0.0.0..0		
JMON	00000037	Listen		
	Local Socket:	0.0.0.0..6715		
	Foreign Socket:	0.0.0.0..0		

もう 1 つ存在する可能性がある制限は、予約済み TCP/IP ポートです。TCP/IP ポートを予約する一般的な場所は、以下の 2 つです。

• PROFILE.TCPIP

これは TCP/IP 開始タスクの PROFILE DD ステートメントによって参照されるデータ・セットで、多くの場合、SYS1.TCPPARMS(TCPPROF) という名前が付いています。

- PORT: 指定されたジョブ名のポートを予約します。
- PORTRANGE: 指定されたジョブ名のポート範囲を予約します。

これらのステートメントの詳細については、「Communications Server IP 構成ガイド」(SC88-8926) を参照してください。

• SYS1.PARMLIB(BPXPRMxx)

- INADDRANYPORT: システムが PORT 0、INADDR_ANY バインドに使用するために予約するポート番号範囲の開始ポート番号を指定します。この値は、CINET (単一ホスト上でアクティブな複数の TCP/IP スタック) にのみ必要です。

- INADDRANYCOUNT: INADDRANYPORT パラメーターで指定したポート番号から始まる、システムが予約するポートの数を指定します。この値は、CINET (単一ホスト上でアクティブな複数の TCP/IP スタック) にのみ必要です。

これらのステートメントの詳細については、「UNIX System Services 計画」(GA88-8639) および「MVS 初期設定およびチューニング解説書」(SA88-8564) を参照してください。

これらの予約済みポートは、**netstat portl** コマンド (TSO または z/OS UNIX) でリストできます。このコマンドは、以下の例のような出力を作成します。

```

MVS TCP/IP NETSTAT CS VxRy          TCPIP Name: TCPIP          17:08:32
Port# Prot User      Flags      Range      IP Address
-----
00007 TCP  MISCSERV DA
00009 TCP  MISCSERV DA
00019 TCP  MISCSERV DA
00020 TCP  OMVS     D
00021 TCP  FTPD1    DA
00025 TCP  SMTP     DA
00053 TCP  NAMESRV  DA
00080 TCP  OMVS     DA
03500 TCP  OMVS     DAR      03500-03519
03501 TCP  OMVS     DAR      03500-03519

```

NETSTAT コマンドの詳細については、「Communications Server IP システム管理者のコマンド」(SC88-9073) を参照してください。

注: **NETSTAT** コマンドは PROFILE.TCPIP 内で定義された情報のみを表示します。これは、BPXPRMxx 定義とオーバーラップします。疑義または問題がある場合は、BPXPRMxx parmlib メンバーを検査して、ここで予約されるポートを確認してください。

アドレス・スペース・サイズ

RSE デーモンは z/OS UNIX Java プロセスであり、機能を実行するために大きな領域サイズを必要とします。したがって、OMVS アドレス・スペースに大きなストレージ限度を設定することが重要です。

始動 JCL の要件

RSE デーモンは、BPXBATSL (この領域サイズは 0 であることが必要です) を使用して JCL によって始動されます。

SYS1.PARMLIB(BPXPRMxx) で設定される制限

SYS1.PARMLIB(BPXPRMxx) で MAXASSIZE を設定してください。これは、デフォルトの OMVS アドレス・スペース (プロセス) 領域サイズを 2G に定義します。これは、許容される最大サイズです。これはシステム全体の限度であるため、すべての z/OS UNIX アドレス・スペースに対してアクティブとなります。これが望ましい値でない場合は、ご使用のセキュリティー・ソフトウェアで Developer for System z 独自の限度を設定できます。

この値は、「MVS システム・コマンド」(GC88-6592) で説明されているように、以下のコンソール・コマンドで検査し、動的に (次回の IPL まで) 設定できます。

1. DISPLAY OMVS,0
2. SETOMVS MAXASSIZE=2G

セキュリティ・プロファイル内に保管される制限

デーモンのユーザー ID OMVS セグメント内で ASSIZEMAX を検査し、2147483647 に設定するか、できれば NONE に設定して SYS1.PARMLIB(BPXPRMxx) 値を使用してください。

RACF を使用している場合は、「*Security Server RACF コマンド言語解説書*」(SA88-8617) で説明されているように、以下の TSO コマンドでこの値を検査および設定できます。

1. LISTUSER userid NORACF OMVS
2. ALTUSER userid OMVS(NOASSIZEMAX)

システム出口によって強制される制限

必ず、システム出口 IEFUSI または IEALIMIT が OMVS アドレス・スペース領域サイズを制御できないようにしてください。これを実現する 1 つの方法は、SYS1.PARMLIB(SMFPRMxx) 内に SUBSYS(OMVS,NOEXITS) をコーディングすることです。

SYS1.PARMLIB(SMFPRMxx) 値は、「MVS システム・コマンド」(GC88-6592) で説明されているように、以下のコンソール・コマンドで検査およびアクティブ化することができます。

1. DISPLAY SMF,0
2. SET SMF=xx

64 ビット・アドレッシングでの制限

SYS1.PARMLIB(SMFPRMxx) 内の MEMLIMIT キーワードでは、64 ビット・タスクが 2 GB 境界より上に割り振ることのできる仮想ストレージの量を制限します。JCL の REGION パラメーターとは異なり、MEMLIMIT=0M は、プロセスが 2 GB 境界より上の仮想ストレージを使用できないことを意味します。

SMFPRMxx に MEMLIMIT を指定しない場合は、デフォルト値 0M が使用されるため、タスクは (31 ビット) 2 GB 境界より下の 2 GB にバインドされます。このデフォルトは z/OS 1.10 で 2G に変更されて、64 ビット・タスクが 4 GB まで使用できるようになりました (2 GB 境界より下の 2 GB および MEMLIMIT で許可される 2 GB 境界より上の 2 GB)。

SYS1.PARMLIB(SMFPRMxx) 値は、「MVS システム・コマンド」(GC88-6592) で説明されているように、以下のコンソール・コマンドで検査およびアクティブ化することができます。

1. DISPLAY SMF,0
2. SET SMF=xx

MEMLIMIT は、JCL で EXEC カードのパラメーターとして指定することもできます。MEMLIMIT パラメーターを指定しない場合は、SMF に定義された値がデフォルトとなります。ただし、REGION=0M を指定した場合は、デフォルトが NOLIMIT となります。

APPC トランザクションおよび TSO コマンド・サービス

TSO コマンド・サービスの APPC バージョンを使用できない場合、2 つの領域で問題が起きることがあります。APPC サーバー・トランザクションの開始と、RSE への接続です。

- APPC のセットアップに関するメッセージが表示されない場合は、システム・ログに RACF メッセージ (メッセージ ID が ICHxxxxx) またはその他の、発行されたコマンドやコマンドを発行したユーザー ID に関連するメッセージがないかどうかを調べます。問題の原因として、よくあるものは以下のとおりです。
 - FEK.SFEKPROC データ・セットに対する読み取り権限を持っていない。
 - TCP/IP がアクティブでないか、誤った DNS 名が付いているか、ネットワークの問題、IP アドレスの誤り、またはその他の原因でシステムが到達不能である (ping 可能でない)。
- APPC のセットアップに関するメッセージが表示され、セットアップの成功を確認するメッセージが表示されない場合は、APPC サーバー・トランザクションを開始できなかったことが考えられます。トランザクション・エラー・ログ (userid.FEKFRSRV.&TPDATE.&TPTIME.LOG) を調べてください。問題の原因としては、以下のようなことが考えられます。
 - TCP/IP スタックが TCPIP のデフォルト名を使用しておらず、SYSTCPD DD カードが設置されていないか、誤ったデータ・セットを指している。
 - サーバーが SYSPROC または SYSTSPRT を割り振ることができなかった。
 - JCL が誤った SYSPROC を指している (SYSPROC は FEK.SFEKPROC を含んでいる必要があります)。
 - サーバーが、MESSAGE_DATA_SET によって参照されているメッセージ (ログ) データ・セットを開くことができなかったか、そのデータ・セットにアクセスできなかった。
 - 十分な数の APPC スケジューラー・イニシエーターが使用可能でない。
 - APPC または ASCH アドレス・スペースがアクティブでない。
 - 使用されるクラス (デフォルト名は「A」) が APPC スケジューラー ASCH に対して定義されていない。
 - システムにデフォルトの OMVS セグメントが存在せず、ユーザーが個人用の OMVS セグメントを持っていないか、そのどちらかのセグメントに定義エラーがある。
 - デフォルトの OMVS セグメントのデフォルト・グループ、またはユーザーのデフォルト・グループに GID 番号がない。

111 ページの『(オプション) TSO コマンド・サービス用の APPC トランザクション』で提供されている REXX は、ISPF パネルを通じて APPC を対話式に管理できる機能を備えているため、APPC の問題の解決に役立つ場合があります。ただし、

このツールを使用してトランザクションを非アクティブにできることに注意してください。その場合、トランザクションは、まだ存在していますが、接続を受け入れなくなります。

以下のリストは、現在のサポート Web サイト (<http://www-306.ibm.com/software/awdtools/rdz/support/>) で入手できる技術情報の抜粋です。詳細については、サポート Web サイトを参照してください。

- APPC 検査が戻りコード 2016 で失敗する - EHOSTNOTFOUND
- APPC 検査が戻りコード 1004 で失敗する - EIBMIUCVERR
- APPC 検査が戻りコード 9 で失敗する - TP 名を認識できない
- APPC 検査が戻りコード 10 で失敗する - TP 使用不可、再試行不可
- APPC 検査が戻りコード 19 で失敗する - パラメーター・エラー
- APPC 検査が戻りコード 20 で失敗する - 製品固有のエラー
- APPC 検査が戻りコード 26 で失敗する - リソース障害
- CEE3501S: モジュール IOSTREAM が検出されなかった
- サーバーを始動できなかった: EDC5129I 指定されたファイルまたはディレクトリがない
- exec/tcp: bind: EDC5111I 許可が拒否された。rsn=744C7246
- サーバーからの応答がなく、次のいずれかのメッセージが表示される
 - IEA995I SYMPTOM DUMP OUTPUT 473 USER COMPLETION CODE=4093 REASON CODE=0000001C (SDSF LOG 内)
 - CEE3512S モジュール libicudata32.0.dll の HFS ロードが失敗した。(CEE3512S An HFS load of module libicudata32.0.dll failed.) システムの戻りコードは 0000000157、理由コードは 0BDF019B でした。(The system return code was 0000000157; the reason code was 0BDF019B.) (CEEDUMP 内)
 - スペースの取得に失敗した (Get Space failed) (クライアントの .log 内)
- コマンド C_CONNECT が使用可能でない
- ホストへの接続時の「FFS サーバーの初期化に失敗しました」エラー・メッセージ
- ホストへの接続時の「EDC5139I 許可されていない操作です (EDC5139I Operation not permitted)」
- MVS ファイルのオープン時の「RSEG1056U FFS サーバーの初期化に失敗しました」

注: このリストは最終的なものではありません。追加の技術情報がないかどうか、サポート Web サイトで確認してください。

各種情報

システム限度

SYS1.PARMLIB(BPXPRMxx) には、複数の Developer for System z クライアントがアクティブなときに到達する可能性がある、z/OS UNIX に関連する多数の限度が定義されています。大部分の BPXPRMxx 値は、**SETOMVS** および **SET OMVS** コンソール・コマンドで動的に変更できます。

BPXPRMxx のいずれかの限度に到達しそうなときに z/OS UNIX でコンソール・メッセージ (BPXI040I) を表示させるには、**SETOMVS LIMMSG=ALL** コンソール・コマンドを使用します。

接続の拒否

各 RSE 接続ごとに複数のプロセスが開始され、それらは永続的にアクティブになります。新規接続は、特にユーザーが同じ UID を共用している場合 (デフォルトの OMVS セグメントを使用している場合など) には、プロセスの量について SYS1.PARMLIB(BPXPRMxx) で設定された限度のために、拒否されることがあります。

- 1 つの UID 当たりの限度は MAXPROCUSER キーワードによって設定され、デフォルト値は 25 です。
- システム全体の限度は MAXPROCSYS キーワードによって設定され、デフォルト値は 200 です。

接続が拒否されるもう 1 つの原因は、アクティブな z/OS アドレス・スペースと z/OS UNIX ユーザー数に課された限度です。

- アドレス・スペース ID (ASID) の最大数は SYS1.PARMLIB(IEASYSxx) 内で MAXUSER キーワードによって定義され、デフォルト値は 255 です。
- z/OS UNIX ユーザー ID (UID) の最大数は SYS1.PARMLIB(BPXPRMxx) 内の MAXUIDS キーワードによって定義され、デフォルト値は 200 です。

必要条件に関する既知の問題

MVS データ・セットのオープンの失敗

TSO コマンド・サービスに APPC を使用する場合に、MVS データ・セットの読み取りと書き込みを行うには、ソケット物理ファイル・システム・ドメインを使用する必要があります。このファイル・システムが正しく定義されていないか、十分な数のソケットを持っていない場合、ロック・マネージャー (FFS) が読み取り/書き込み要求を処理できないことがあります。ffs*.log ファイルには、次のようなメッセージが表示されます。

- エラー 127 ソケット・ペアの取得 - ポートを 0 に設定。(Error 127 getting socket pair - setting port to 0.)
- UNIX ドメイン内にソケットを作成できない。エラーは、「アドレス・ファミリーはサポートされていません。」

SYS1.PARMLIB(BPXPRMxx) メンバーに以下のステートメントが含まれていることを確認してください。

```
FILESYSTYPE TYPE(UDS) ENTRYPOINT(BPXTUINT)
NETWORK DOMAINNAME(AF_UNIX)
      DOMAINNUMBER(1)
      MAXSOCKETS(2000)
      TYPE(UDS)
```

TSO コマンド・サービスに APPC を使用している場合、リゾルバー構成ファイルが欠落しているか、不完全であるために、TCP/IP リゾルバーがホスト・アドレスを正しく解決できないということも、この問題の原因として考えられます。この問題を明確に示すものは、lock.log 内の以下のメッセージです。

```
clientip(0.0.0.0) <> callerip(<host IP address>)
```

117 ページの『第 7 章 インストール検査』で説明されている方法に従って、fekfivpt TCP/IP IVP を実行してください。出力のリゾルバー構成セクションは、以下のサンプルのようになります。

```
Resolver Trace Initialization Complete -> 2008/07/02 13:11:54.745964
```

```
res_init Resolver values:
Global Tcp/Ip Dataset = None
Default Tcp/Ip Dataset = None
Local Tcp/Ip Dataset = /etc/resolv.conf
Translation Table = Default
UserId/JobName = USERID
Caller API = LE C Sockets
Caller Mode = EBCDIC
```

「Local Tcp/Ip Dataset」が示すファイル (データ・セット) 内の定義が正しいことを確認してください。

このフィールドは、IP リゾルバー・ファイルに (z/OS UNIX 検索順序を使用して) デフォルト名を使用していない場合にはブランクになります。その場合は、次のステートメントを rsed.envvars に追加します。ここで、<resolver file> または <resolver data> は IP リゾルバー・ファイルの名前を表しています。

```
RESOLVER_CONFIG=<resolver file>
```

または

```
RESOLVER_CONFIG='<resolver data set>'
```

Host Connect Emulator

- Host Connect Emulator は、ホストへの接続に RSE サーバーでなく、TN3270 Telnet を使用します。
- セキュア Telnet (SSL) を使用し、既知の CA による署名がない証明書を使用して作業している場合、すべてのクライアントは、トラステッド CA の Host Connect Emulator リストに CA 証明書を追加する必要があります。
- SNA 機能拡張を使用不可にするために、TCP/IP の TELNETPARMS の NOSNAEXT オプションが必要になる場合があります。NOSNAEXT を指定した場合、TN3270 Telnet サーバーは競合の解決と SNA センス機能についてネゴシエーションを行いません。

第 10 章 セキュリティーに関する考慮事項

Developer for System z では、メインフレーム以外のワークステーション上にいるユーザーがメインフレームにアクセスできます。このため、接続要求の妥当性検査、ホストとワークステーション間のセキュアな通信の提供、およびアクティビティーの許可と監査が、製品構成の重要な側面となります。

Developer for System z サーバーが使用するセキュリティー・メカニズムは、そのサーバーが存在するファイル・システムがセキュアであることに依存しています。つまり、信頼されたシステム管理者のみがプログラム・ライブラリーと構成ファイルを更新できる状態でなければなりません。

この章では、以下のトピックについて説明します。

- 170 ページの『認証方式』
- 171 ページの『接続セキュリティー』
- 173 ページの『TCP/IP ポート』
- 175 ページの『PassTicket の使用』
- 176 ページの『監査ロギング』
- 177 ページの『JES セキュリティー』
- 181 ページの『SSL 暗号化通信』
- 182 ページの『X.509 証明書を使用したクライアント認証』
- 186 ページの『Port Of Entry (POE) 検査』
- 187 ページの『CICSTS セキュリティー』
- 187 ページの『SCLM セキュリティー』
- 188 ページの『Developer for System z 構成ファイル』
- 189 ページの『セキュリティー定義』

注: リモート・システム・エクスプローラー (RSE) は、クライアントをホストに接続するなどのコア・サービスを提供し、次の 2 つの論理エンティティーから構成されています。

- RSE デーモン。これは接続セットアップを管理し、開始タスクとして開始されるか、長時間実行ユーザー・ジョブとして開始されます。
- RSE サーバー。これは個々のクライアント要求を処理し、RSE デーモンによって、1 つ以上の子プロセス内のスレッドとして開始されます。

Developer for System z の基本的な設計概念については、205 ページの『第 11 章 Developer for System z について』を参照してください。

認証方式

Developer for System z は、接続時にクライアントが提供するユーザー ID を認証するために、複数の方法をサポートしています。

- ユーザー ID およびパスワード
- ユーザー ID およびワンタイム・パスワード
- X.509 証明書

クライアントが提供した認証データは、初期の接続セットアップ時に 1 回だけ使用されることに注意してください。ユーザー ID が認証された後、認証を必要とするすべてのアクションには、そのユーザー ID と自己生成された PassTicket が使用されます。

ユーザー ID およびパスワード

クライアントは接続時に、ユーザー ID とそれに一致するパスワードを提供します。そのユーザー ID とパスワードは、使用しているセキュリティ製品でのユーザーの認証に使用されます。

ユーザー ID およびワンタイム・パスワード

ワンタイム・パスワードは、固有のトークンを基に、サード・パーティー製品で生成できます。ワンタイム・パスワードでは、ユーザーが知らないうちに固有のトークンをコピーして使用することはできないため、セキュリティのセットアップが向上し、またパスワードをインターセプトしても、それは一回限り有効であるため役に立ちません。

クライアントは接続時に、ユーザー ID とワンタイム・パスワードを提供します。このパスワードは、サード・パーティーが提供するセキュリティ出口で、ユーザー ID の認証に使用されます。このセキュリティ出口では、通常の処理時に認証要求を満たすために使用された PassTicket を無視することが見込まれます。PassTicket は、ご使用のセキュリティ・ソフトウェアによって処理する必要があります。

X.509 証明書

サード・パーティーは、ユーザーの認証に使用できる 1 つ以上の X.509 証明書を提供できます。X.509 証明書を機密保護機能のあるデバイスに保管すると、セキュアなセットアップとユーザーにとっての操作性（ユーザー ID やパスワードが不要であること）を同時に実現できます。

接続時に、クライアントは選択された証明書と選択された拡張（オプション）を提供します。これを使用して、ご使用のセキュリティ製品でユーザー ID の認証が行われます。

この認証方式は RSE デーモン接続方式によってのみサポートされており、SSL を使用可能にする必要があることに注意してください。

JES ジョブ・モニターの認証

クライアント認証は、クライアントの接続要求の一環として、RSE デーモン (または REXEC/SSH) によって行われます。ユーザーが認証されると、JES ジョブ・モニターへの自動ログオンも含め、その後のすべての認証要求には、自己生成された PassTicket が使用されます。

JES ジョブ・モニターは、RSE によって提供されたユーザー ID と PassTicket の妥当性検査を行うためには、PassTicket の評価を許可されている必要があります。これは、以下のことを意味します。

- ロード・モジュール FEJMON (デフォルトではロード・ライブラリー FEK.SFEKAUTH に置かれています) に APF 許可があることが必要です。
- RSE と JES ジョブ・モニターの両方が、同じアプリケーション ID (APPLID) を使用する必要があります。デフォルトでは、どちらのサーバーも FEKAPPL を APPLID として使用しますが、これは変更でき、そのためには、RSE の場合は rsed.envvars 内で、JES ジョブ・モニターの場合は FEJJCNFG 内で、それぞれ APPLID ディレクティブを使用します。

注: 以前のクライアント (バージョン 7.0 以前) は、JES ジョブ・モニターと直接通信します。これらの接続では、ユーザー ID とパスワードによる認証方式のみがサポートされています。

接続セキュリティ

さまざまなレベルの通信セキュリティが RSE によってサポートされており、RSE は、クライアントと Developer for System z サービスの間のすべての通信を制御します。

- 外部 (クライアント/ホスト) 通信を、指定したポートだけに制限できます。このフィーチャーは、デフォルトでは使用不可に設定されます。
- 外部 (クライアント/ホスト) 通信を SSL で暗号化できます。このフィーチャーは、デフォルトでは使用不可に設定されます。
- 信頼できる TCP/IP アドレスのみにホスト・アクセスを許可できるようにするために、Port Of Entry (POE) 検査を使用できます。このフィーチャーは、デフォルトでは使用不可に設定されます。

指定したポートのみに限定した外部通信

システム・プログラマーは、RSE サーバーがクライアントと通信できるポートを指定できます。デフォルトでは、使用可能な任意のポートが使用されます。このポート範囲は、RSE デーモン・ポートとは関係ありません。

ポートの使用法を理解しやすいように、以下で RSE の接続プロセスを簡単に説明します。

1. クライアントは、ホスト・ポート 4035、RSE デーモンに接続します。
2. RSE デーモンは、RSE サーバー・スレッドを作成します。
3. RSE サーバーは、クライアントが接続するホスト・ポートを開きます。このポートの選択はユーザーが構成でき、クライアント上でサブシステム・プロパティ

ー・タブによって構成するか (これはお勧めできません)、rsed.envvars 内の _RSE_PORTRANGE 定義を通じて構成します。

4. RSE デーモンは、クライアントにポート番号を返します。
5. クライアントは、ホスト・ポートに接続します。

注: このプロセスは、109 ページの『(オプション) REXEC (または SSH) の使用』で説明されている REXEC/SSH を使用した (オプションの) 代替接続方式とほとんど同じです。

SSL を使用した通信暗号化

RSE を通過するすべての外部 Developer for System z データ・ストリームを、Secure Sockets Layer (SSL) によって暗号化できます。SSL の使用は、181 ページの『SSL 暗号化通信』での説明のように、ssl.properties 構成ファイル内の設定によって制御されます。

クライアント上の Host Connect Emulator は、ホスト上の TN3270 サーバーに接続します。SSL の使用は TN3270 によって制御されます。これについては、「*Communications Server IP 構成ガイド*」(SC88-8926) に説明があります。

Application Deployment Manager クライアントは、CICS TS Web サービスまたは RESTful インターフェースを使用して、Application Deployment Manager ホスト・サービスを起動します。SSL の使用は、CICS TS によって制御されます。これについては、「*RACF Security Guide for CICS TS*」に説明があります。

Port Of Entry 検査

Developer for System z は Port Of Entry (POE) 検査をサポートしています。これにより、ホストは信頼できる TCP/IP アドレスにのみアクセスできます。POE の使用は、186 ページの『Port Of Entry (POE) 検査』で説明されているように、セキュリティー・ソフトウェア内の特定のプロファイルの定義と、rsed.envvars 内の enable.port.of.entry ディレクティブによって制御されます。

POE をアクティブにすると、POE 検査をサポートしている他の TCPIP アプリケーション (INETD など) に影響が出ることに注意してください。

TCP/IP ポート

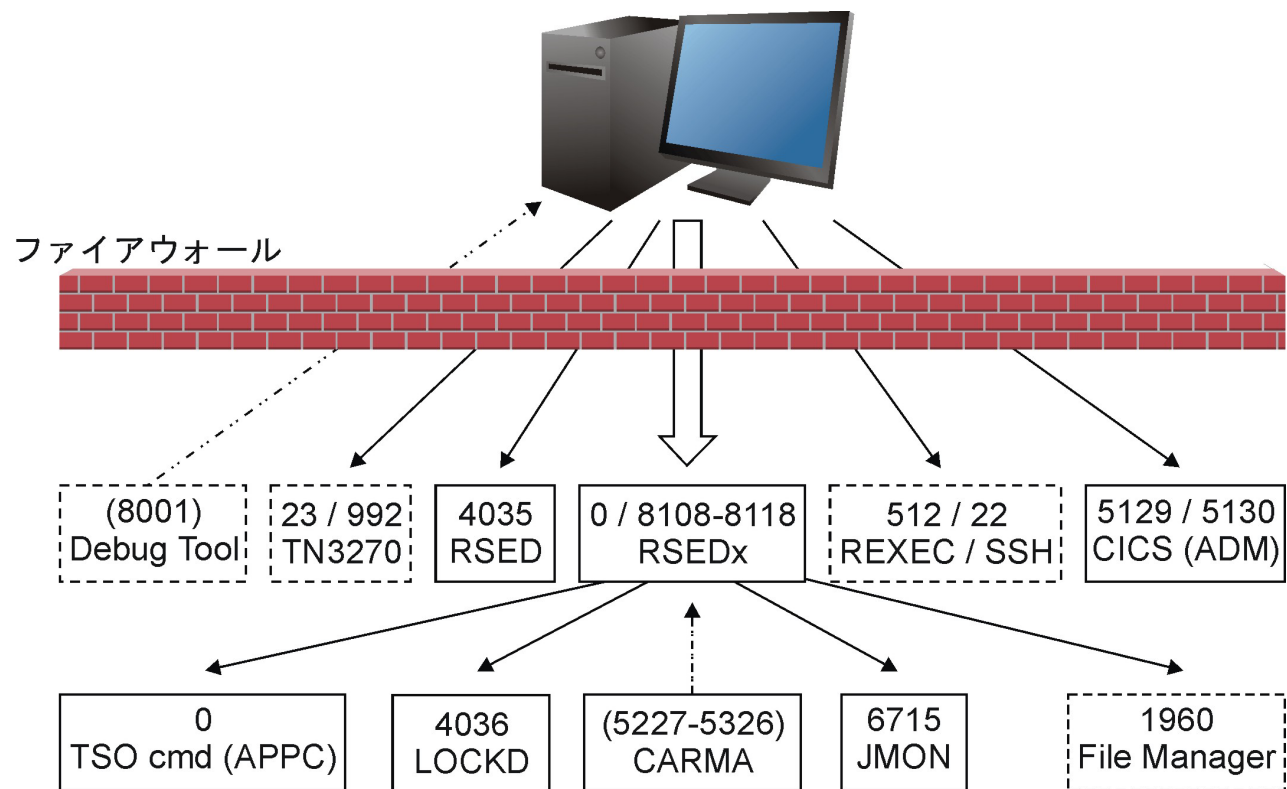


図 40. TCP/IP ポート

図 40 は、Developer for System z で使用できる TCP/IP ポートを示しています。矢印は、バインドの実行元 (矢印の先) と接続元を示しています。

外部通信

z/OS ホストを保護しているファイアウォールに対して、以下のポートを定義してください。これらのポートは、クライアント/ホスト通信 (tcp プロトコルを使用) に使用されるためです。

- クライアント/ホスト通信セットアップ用の RSE デーモン、デフォルト・ポート 4035。このポート上の通信は、SSL を使用して暗号化できます。
- クライアント/ホスト通信用の RSE サーバー。デフォルトでは、使用可能な任意のポートを使用できますが、これは `rsed.envvars` 内の `_RSE_PORTRANGE` 定義によって、指定する範囲に制限できます。このポート上の通信は、SSL を使用して暗号化できます。
- (オプション) z/OS UNIX サブプロジェクトのリモート (ホスト・ベースの) アクション用の以下のいずれかの INETD サービス。
 - REXEC (z/OS UNIX バージョン)、デフォルト・ポート 512。
 - SSH (z/OS UNIX バージョン)、デフォルト・ポート 22。このポート上の通信は、SSL を使用して暗号化されます。
- (オプション) Host Connect Emulator 用の TN3270 Telnet サービス、デフォルト・ポート 23。通信は、SSL を使用して暗号化できます (デフォルト・ポート

992)。TN3270 Telnet サービスに割り当てられるデフォルト・ポートは、ユーザーが暗号化の使用を選択するかどうかによって決まります。

- (オプション) Application Deployment Manager 用の、以下の CICSTS アプリケーション・インターフェース (いずれかまたは両方)
 - RESTful インターフェース、デフォルト・ポート 5130。
 - Web サービス・インターフェース、デフォルト・ポート 5129。このポート上の通信は、SSL を使用して暗号化できます。

注:

- 以前のクライアント (バージョン 7.0 以前) は、JES ジョブ・モニターと直接通信します。デフォルト・ポート 6715。
- COBOL、PL/I、またはアセンブラーのリモート・デバッグ・セッションのときは、IBM Debug Tool for z/OS が起動されます。この製品は、クライアントと直接通信します。この通信はホスト上で開始され、クライアント上のポート 8001 に接続します。

内部通信

いくつかの Developer for System z ホスト・サービスは別のスレッドまたはアドレス・スペースで実行され、TCP/IP ソケットを通信メカニズムとして使用します。これらすべてのサービスは、クライアントとの通信に RSE を使用し、データ・ストリームをホストだけに限定します。一部のサービスでは、使用可能な任意のポートが使用され、それ以外のサービスでは、使用されるポートまたはポート範囲をシステム・プログラマーが選択できます。

- JES 関連サービスの JES ジョブ・モニター、デフォルト・ポート 6715。このポートは、FEJJCNFG 構成メンバーの中で設定できます。
- データ・セット・ロック関連サービスのロック・デーモン、デフォルト・ポート 4036。このポートは、rsed.envvars 構成メンバーの中で設定できます。
- (オプション) IBM File Manager と対話するための File Manager Integration、デフォルト・ポート 1960。
- (オプション) CARMA 通信、デフォルト・ポート範囲 5227 から 5326 (100 ポート)。このポート範囲は、CRASRV.properties 構成ファイルの中で設定できます。
- (オプション) APPC バージョンの TSO コマンド・サービスは、使用可能な任意のソケットを使用してロック・マネージャー (これは、クライアント用の MVS データ・セットをキューに入れます) と通信します。使用する特定のポート範囲を設定することはできません。

注: 以前のクライアント (バージョン 7.0 以前) は、JES ジョブ・モニター・サーバーと直接通信します。デフォルト・ポート 6715。

CARMA と TCP/IP ポート

ほとんどの場合、サーバーは RSE デーモンの場合と同様に、ポートにバインドし、接続要求を listen します。しかし CARMA は別の方法を使用します。これは、クライアントが接続要求を開始した時点で CARMA サーバーがまだアクティブでないためです。

クライアントから接続要求が送信されると、RSE スレッド・プール内でユーザー・スレッドとしてアクティブとなっている CARMA マイナーは、CRASRV.properties 構成ファイルに指定されている範囲から空いているポートを見つけ、そのポートにバインドします。次にこのマイナーは、CARMA サーバーを始動してポート番号を渡します。これによって、サーバーは接続先のポートを認識します。サーバーが接続されると、クライアントはサーバーに要求を送信して結果を受信できるようになります。

したがって TCP/IP の観点では、RSE (CARMA マイナー経由) がポートにバインドするサーバーであり、CARMA サーバーがそのポートに接続するクライアントです。

PassTicket の使用

ログオン後、PassTicket を使用して RSE サーバー内のスレッドのセキュリティが確立されます。このフィーチャーを使用不可能にすることはできません。PassTicket は、有効期間が約 10 分のシステム生成パスワードです。生成される PassTicket は、DES 暗号化アルゴリズム、ユーザー ID、アプリケーション ID、時刻と日付のスタンプ、および秘密鍵に基づいています。この秘密鍵は 64 ビットの数値 (16 個の 16 進文字) で、これは、使用するセキュリティ・ソフトウェアに対して定義されている必要があります。

PassTicket の使用法を理解しやすいように、以下で RSE のセキュリティ・プロセスを簡単に説明します。

1. クライアントは、ホスト・ポート 4035、RSE デーモンに接続します。
2. RSE デーモンはクライアントが提示した資格情報を使用して、クライアントを認証します。
3. RSE デーモンは、固有のクライアント ID と RSE サーバー・スレッドを作成します。
4. RSE サーバーは PassTicket を生成し、その PassTicket をパスワードとして使用して、クライアント用のセキュリティ環境を作成します。
5. クライアントは、RSE デーモンが返したホスト・ポートに接続します。
6. RSE サーバーはクライアント ID を使用して、クライアントの妥当性を検査します。
7. RSE サーバーは新規に作成された PassTicket を、その後のパスワードを必要とするすべてのアクションにパスワードとして使用します。

クライアントの実際のパスワードは、初期認証後は不要になります。SAF 準拠のセキュリティ製品は、PassTicket と正規のパスワードのどちらも評価できるからです。RSE サーバーはパスワードが必要になるたびに、PassTicket を生成して使用します。その結果、PassTicket はクライアントの (一時的に) 有効なパスワードになります。

PassTicket を使用すると、RSE はユーザー固有のセキュリティ環境を任意にセットアップでき、すべてのユーザーの ID とパスワードをテーブルに保管する (これはセキュリティを脅威にさらす可能性があります) 必要がなくなります。また、X.509 証明書など、再使用可能なパスワードを使用しないクライアント認証方式も使用できるようになります。

PassTicket を使用できるようにするには、APPL クラスと PTKTDATA クラスのセキュリティ・プロファイルが必要です。これらのプロファイルはアプリケーション固有のものであるため、現在使用しているシステムのセットアップに影響を及ぼしません。

PassTicket がアプリケーション固有のものであることは、RSE と JES ジョブ・モニターの両方が、同じアプリケーション ID (APPLID) を使用する必要があることを意味します。デフォルトでは、どちらのサーバーも FEKAPPL を APPLID として使用しますが、これは変更でき、そのためには、RSE の場合は `rsed.envvars` 内で、JES ジョブ・モニターの場合は `FEJJCNF` 内で、それぞれ APPLID ディレクティブを使用します。

OMVSAPPL はアプリケーション ID として使用しないでください。これは、大部分の z/OS UNIX アプリケーションの秘密鍵を公開するからです。また、デフォルトの MVS アプリケーション ID (MVS の直後にシステムの SMF ID を続けたもの) も使用しないでください。これは、大部分の MVS アプリケーション (ユーザー・バッチ・ジョブを含む) の秘密鍵を公開するからです。

重要: PassTicket が正しくセットアップされていないと、クライアントの接続要求は失敗します。
--

監査ロギング

Developer for System z は、RSE デーモンによって管理されるアクションの監査ロギングをサポートしています。監査ログは、CSV (コンマ区切り値) 形式のテキスト・ファイルとしてデーモン・ログ・ディレクトリーに保管されます。

監査制御

46 ページの『`_RSE_JAVAOPTS` での追加 Java 始動パラメーターの定義』で説明されているように、`rsed.envvars` 内の複数のオプションが監査機能に影響を及ぼします。

- 監査機能は、`enable.audit.log` オプションによって使用可能/使用不可にされます。
- 監査のデフォルトは、`audit.*` オプションによって制御されます。
- 監査ログ・ファイルのロケーションは、`daemon.log` オプションによって制御されます。
- 監査ログの書き込みに使用されるコード・ページは、35 ページの『`rsed.envvars`、RSE 構成ファイル』で説明しているように、`_RSE_HOST_CODEPAGE` ディレクティブによって制御されます。

modify switch オペレーター・コマンドを使用して、133 ページの『第 8 章 オペレーター・コマンド』の説明にあるように、新しい監査ログ・ファイルに手動で切り替えることができます。

監査ログ・ファイルを保持しているファイル・システムのフリー・スペースが少なくなった場合は、コンソールへ警告メッセージが送信されます。このコンソール・メッセージ (FEK103E) は、スペース不足の問題が解決されるまで定期的に繰り返さ

れます。RSE が生成するコンソール・メッセージのリストについては、140 ページの『コンソール・メッセージ』を参照してください。

監査データ

新しい監査ログ・ファイルは、あらかじめ定められた時間が経過した後、または **modify switch** オペレーター・コマンドが発行されたときに開始されます。古いログ・ファイルは、`audit.log.yyyymmdd.hhmmss` として保存されます。ここで、`yyymmdd.hhmmss` は、そのログが閉じられたときの日付/タイム・スタンプです。ファイルへ割り当てられたシステム日付/タイム・スタンプは、ログ・ファイルの作成を示しています。これら 2 つの日付の組み合わせが、この監査ログ・ファイルの記録期間を示しています。

以下のアクションがログとして記録されます。

- システム・アクセス (接続、切断)
- JES スプール・アクセス (実行依頼、表示、保留、保留解除、キャンセル、ページ)
- データ・セット・アクセス (読み取り、書き込み、作成、削除、名前変更、圧縮、マイグレーション、再呼び出し)
- TSO コマンドの実行

ログに記録された各アクションは、CSV (コンマ区切り値) 形式で (日付/タイム・スタンプ付きで) 保管されます。CSV 形式は、自動化ツールまたはデータ分析ツールで読み取ることができます。

監査ログ・ファイルには許可ビット・マスク 640 (-rw-r-----) があります。これは、所有者 (RSE デーモン `z/OS UNIX uid`) が読み取りおよび書き込みアクセス権を持ち、所有者の (デフォルトの) グループが読み取りアクセス権を持つことを意味しています。それ以外のアクセスの試みは、すべて拒否されます。ただし、スーパーユーザー (UID 0) または UNIXPRIV クラスの `SUPERUSER.FILESYS` プロファイルに対する十分な権限を持つユーザーが試みた場合は除きます。

JES セキュリティー

Developer for System z では、クライアントは JES ジョブ・モニターを通じて JES スプールにアクセスできます。サーバーは基本的なアクセス制限を行い、この制限は、ご使用のセキュリティ製品の標準スプール・ファイル保護フィーチャーによって拡張できます。スプール・ファイルに対するオペレーター・アクション (「保留」、「保留解除」、「キャンセル」、および「ページ」) は EMCS コンソールを通じて行われ、このコンソールについて、条件付きの許可をセットアップする必要があります。

ジョブに対するアクション - ターゲットの制限

JES ジョブ・モニターは、Developer for System z ユーザーに JES スプールへのフル・オペレーター・アクセスを提供するわけではありません。保留、保留解除、キャンセル、ページの各コマンドのみが使用可能で、しかも、デフォルトでは、そのユーザーが所有しているスプール・ファイルの場合だけです。コマンドは、クライアント・メニュー構造で該当するオプションを選択することによって発行されます。

(コマンド・プロンプトはありません)。コマンドのスコープは、セキュリティー・プロファイルを使用してコマンドの使用対象となるジョブを定義することにより、広げることができます。

SDSF の **SJ** アクション文字と同じように、JES ジョブ・モニターは「JCL の表示」コマンドもサポートしています。このコマンドは、選択されたジョブ出力を作成した JCL を取り出して、エディター・ウィンドウに表示します。JES ジョブ・モニターは JES から JCL を取り出せるので、元の JCL メンバーを容易に見つけることができない状況で役立ちます。

表 21. JES ジョブ・モニターのコンソール・コマンド

アクション	JES2	JES3
保留	\$Hx(jobid) x = {J、S、または T}	*F,J=jobid,H
保留解除	\$Ax(jobid) x = {J、S、または T}	*F,J=jobid,R
キャンセル	\$Cx(jobid) x = {J、S、または T}	*F,J=jobid,C
ページ	\$Cx(jobid),P x = {J、S、または T}	*F,J=jobid,C
JCL の表示	適用外	適用外

表 21 にリストした使用可能な JES コマンドは、デフォルトでは、そのユーザーが所有しているジョブだけに制限されます。これは、30 ページの『FEJJCNFG、JES ジョブ・モニター構成ファイル』で説明されているように、LIMIT_COMMANDS ディレクティブを使用して変更できます。

表 22. LIMIT_COMMANDS コマンドの許可のマトリックス

LIMIT_COMMANDS	ジョブ所有者	
	ユーザー	その他
USERID (デフォルト)	許可される	許可されない
LIMITED	許可される	セキュリティー・プロファイルによって明示的に許可された場合にのみ許可される
NOLIMIT	許可される	セキュリティー・プロファイルによって許可された場合、または JESSPOOL クラスがアクティブでない場合は許可される

JES は、JESSPOOL クラスを使用して SYSIN/SYSOUT データ・セットを保護します。SDSF と同じように、JES ジョブ・モニターは JESSPOOL クラスの用途を拡張し、ジョブ・リソースの保護も行います。

LIMIT_COMMANDS が USERID でない場合、JES ジョブ・モニターは、次の表に示すように、JESSPOOL クラス内の関連するプロファイルに対する権限を照会します。

表 23. 拡張 JESSPOOL プロファイル

コマンド	JESSPOOL プロファイル	必要なアクセス権
保留	nodeid.userid.jobname.jobid	ALTER
保留解除	nodeid.userid.jobname.jobid	ALTER
キャンセル	nodeid.userid.jobname.jobid	ALTER
ページ	nodeid.userid.jobname.jobid	ALTER
JCL の表示	nodeid.userid.jobname.jobid.JCL	READ

前記の表で、以下のように置き換えてください。

nodeid	ターゲット JES サブシステムの NJE ノード ID
userid	ジョブ所有者のローカル・ユーザー ID
jobname	ジョブの名前
jobid	JES ジョブ ID

JESSPOOL クラスがアクティブでない場合、LIMIT_COMMANDS の LIMITED および NOLIMIT 値の動作は、33 ページの表 9 の説明のように異なります。JESSPOOL がアクティブの場合、動作は同じです。クラスは、デフォルトでは、プロファイルが定義されていないければ許可を拒否するからです。

ジョブに対するアクション - 実行の制限

許可されたターゲットを指定した後の、JES スプール・コマンド・セキュリティの第 2 フェーズには、オペレーター・コマンドを実際に実行するために必要な許可が含まれます。この実行許可は、z/OS および JES のセキュリティ検査によって行われます。

「JCL の表示」は、他の JES ジョブ・モニター・コマンド（「保留」、「保留解除」、「キャンセル」、および「ページ」）のようなオペレーター・コマンドでないため、以下の制限が（それ以上のセキュリティ検査が存在しないので）適用されないことに注意してください。

JES ジョブ・モニターは、ユーザーが要求したすべての JES オペレーター・コマンドを、拡張 MCS (EMCS) コンソールを通じて発行します。このコンソールの名前は、30 ページの『FEJJCNFG、JES ジョブ・モニター構成ファイル』の説明にあるように、CONSOLE_NAME ディレクティブによって制御されます。

このセットアップでは、セキュリティ管理者は OPERCMDS クラスおよび CONSOLE クラスを使用して、詳細なコマンド実行権限を定義できます。

- EMCS コンソールを使用するユーザーは、OPERCMDS クラス内の MVS.MCSOPER.console-name プロファイルに対する READ 権限を（最低限）持っている必要があります。プロファイルが定義されていない場合、システムは権限要求を認可することに注意してください。
- JES オペレーター・コマンドを実行するユーザーは、OPERCMDS クラス内の JES%.**（または、それ以上の詳細な）プロファイルに対する十分な権限を持って

いる必要があります。プロファイルが定義されていないか、OPERCMDS クラスがアクティブでない場合、JES はコマンドの実行に失敗します。

- セキュリティー管理者は、**PERMIT** 定義で **WHEN(CONSOLE(JMON))** を指定することにより、ユーザーがオペレーター・コマンドを実行するときに、JES ジョブ・モニターの使用を必須とすることができます。このセットアップを機能させるには、CONSOLE クラスをアクティブにする必要があります。CONSOLE クラスがアクティブであれば十分であることに注意してください。EMCS コンソールの有無について、プロファイルが検査されることはありません。

TSO セッションから JMON コンソールを作成することによって JES ジョブ・モニター・サーバーの ID を装うことは、セキュリティー・ソフトウェアによって防止されます。コンソールを作成できる場合でも、(JES ジョブ・モニターと TSO とでは) 入り口点が異なります。この資料で説明されているとおりにセキュリティーがセットアップされており、ユーザーが他の手段によって JES コマンドに対する権限を持っていない場合は、そのコンソールから発行された JES コマンドはセキュリティー検査で不合格になります。

JES ジョブ・モニターは、コマンドを実行する必要があるとき、コンソール名が既に使用されていると、コンソールを作成できないことに注意してください。これを防止するために、システム・プログラマーは JES ジョブ・モニター構成ファイル内で **GEN_CONSOLE_NAME=ON** ディレクティブを設定でき、セキュリティー管理者は TSO ユーザーがコンソールを作成しないようにセキュリティー・プロファイルを定義できます。以下のサンプル RACF コマンドは、(許可されたユーザーを除いて) 誰も TSO または SDSF コンソールを作成できないようにします。

- **RDEFINE TSOAUTH CONSOLE UACC(NONE)**
- **PERMIT CONSOLE CLASS(TSOAUTH) ACCESS(READ) ID(#userid)**
- **RDEFINE SDSF ISFCMD.ODSP.ULOG.* UACC(NONE)**
- **PERMIT ISFCMD.ODSP.ULOG.* CLASS(SDSF) ACCESS(READ) ID(#userid)**

注: これらのオペレーター・コマンドの許可がない場合でも、ユーザーは、これらのリソースを保護できるプロファイル (JESINPUT、JESJOBS、および JESSPOOL クラス内のプロファイルなど) に対して十分な権限を持っていれば、JES ジョブ・モニターを通じてジョブを実行依頼し、ジョブ出力を読み取ることができます。

オペレーター・コマンド保護の詳細については、「*Security Server RACF セキュリティー管理者のガイド*」(SA88-8613) を参照してください。

スプール・ファイルへのアクセス

JES ジョブ・モニターは、デフォルトでは、すべてのスプール・ファイルへの参照アクセスを許可します。これは、30 ページの『FEJJCNFG、JES ジョブ・モニター構成ファイル』で説明されているように、**LIMIT_VIEW** ディレクティブを使用して変更できます

表 24. **LIMIT_VIEW** のブラウズ権限のマトリックス

LIMIT_VIEW	ジョブ所有者	
	ユーザー	その他

表 24. LIMIT_VIEW のブラウザ権限のマトリックス (続き)

USERID	ジョブ所有者	
	許可される	許可されない
NOLIMIT (デフォルト)	許可される	セキュリティ・プロファイルによって許可された場合、または JESSPOOL クラスがアクティブでない場合は許可される

ユーザーを JES スプール上のそのユーザー自身のジョブだけに制限するには、JES ジョブ・モニター構成ファイル FEJJCNF で "LIMIT_VIEW=USERID" ステートメントを定義します。すべてのジョブではありませんが、より広範囲のジョブへのアクセスをユーザーが必要とする場合は、使用しているセキュリティ製品の標準スプール・ファイル保護フィーチャー、例えば、JESSPOOL クラスなどを使用します。

さらに保護を定義するときは、JES ジョブ・モニターがスプールへのアクセスに SAPI (SYSOUT アプリケーション・プログラム・インターフェース) を使用することに留意してください。これは、ユーザーが単にブラウザ機能を使用するだけでも、最低限、スプール・ファイルに対する UPDATE アクセス権が必要になることを意味します。この必要条件は、z/OS 1.7 (JES3 の場合は z/OS 1.8) 以上を実行している場合には適用されません。この場合、ブラウザ機能を使用するには、READ 権限で十分です。

JES スプール・ファイルの保護の詳細については、「*Security Server RACF セキュリティ管理者のガイド*」(SA88-8613) を参照してください。

SSL 暗号化通信

外部 (クライアント/ホスト) 通信を SSL (Secure Sockets Layer) で暗号化できます。このフィーチャーは、デフォルトでは使用不可に設定され、100 ページの『(オプション) RSE SSL 暗号化』の説明にあるように、ssl.properties 内の設定によって制御されます。

RSE デーモンおよび RSE サーバーは、両者間のアーキテクチャーの違いから、証明書の保管に関して異なるメカニズムをサポートしています。これは、RSE デーモンと RSE サーバーの両方に SSL 定義と証明書が必要であることを意味しています。RSE デーモンと RSE サーバーが同じ証明書管理方式を使用する場合は、共用証明書を使用できます。

表 25. SSL 証明書の保管メカニズム

証明書ストレージ	作成者および管理者	RSE デーモン	RSE サーバー
鍵リング	SAF 準拠のセキュリティ製品	サポート	サポート
鍵データベース	z/OS UNIX の gskkyman	サポート	/
鍵ストア	Java の keytool	/	サポート

注: SAF 準拠の鍵リングは、証明書の管理に推奨される方式です。

SAF 準拠の鍵リングでは、証明書の秘密鍵が、セキュリティー・データベースに保管されるか、または System z 暗号化ハードウェアとのインターフェースである ICSF (Integrated Cryptographic Service Facility) を使用して保管されます。

ICSF は、非 ICSF の秘密鍵管理よりもセキュアなソリューションであるため、デジタル証明書に関連する秘密鍵の保管には ICSF の使用をお勧めします。ICSF では、確実に、秘密鍵が ICSF マスター・キーで暗号化され、かつその秘密鍵へのアクセスが CSFKEYS および CSFSERV セキュリティー・クラスの一般リソースによって制御されます。さらに、ICSF はハードウェア Cryptographic Coprocessor (暗号化コプロセッサ) を使用するため、操作上のパフォーマンスが向上します。

RSE デーモンは、System SSL の機能を使用して SSL 暗号化通信を管理します。これは SYS1.SIEALNKE が、ご使用のセキュリティー・ソフトウェアによってプログラム制御されることが必要で、LINKLIST または rsed.envvars 内の STEPLIB ディレクティブを介して RSE から使用可能でなければならないことを意味しています。

RSE デーモンまたは RSE サーバーに SAF 準拠の鍵リングが使用されている場合は、RSE ユーザー ID (以下のサンプル・コマンドでは STCRSE) に、鍵リングとそれに関連する証明書にアクセスするための権限が必要です。

- RDEFINE FACILITY IRR.DIGTCERT.LIST UACC(NONE)
- RDEFINE FACILITY IRR.DIGTCERT.LISTRING UACC(NONE)
- PERMIT IRR.DIGTCERT.LIST CLASS(FACILITY) ACCESS(READ) ID(stcrse)
- PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) ACCESS(READ) ID(stcrse)
- SETROPTS RACLIST(FACILITY) REFRESH

Developer for System z 用に SSL をアクティブにする方法について詳しくは、315 ページの『付録 A. SSL および X.509 認証のセットアップ』を参照してください。

X.509 証明書を使用したクライアント認証

RSE デーモンは、X.509 証明書で自分自身を認証するユーザーをサポートしています。この機能の場合、SSL 暗号化通信の使用は前提条件です。この機能は、SSL で使用される証明書でのホスト認証を拡張したものです。

RSE デーモンは、クライアント証明書の妥当性を検査することによって、クライアント認証プロセスを開始します。検査される重要な点は、証明書が有効である日付、および、証明書の署名に使用された認証局 (CA) の信頼性などです。オプションとして、(サード・パーティーの) 証明書失効リスト (CRL) を調べることもできます。

RSE デーモンが証明書の妥当性を検査した後、認証のために証明書が処理されます。証明書は、rsed.envvars ディレクティブの enable.certificate.mapping が false に設定されている場合を除き、使用しているセキュリティー製品へ渡され、その時点で RSE デーモンは認証を行います。

認証プロセスが成功した場合、そのユーザー ID をこのセッションに使用することが決定され、その後、RSE デーモンはユーザー ID をテストし、RSE デーモンが稼働しているホスト・システム上で使用可能であるかどうかを確認します。

最終検査 (X.509 証明書だけでなく、すべての認証メカニズムで行われます) では、ユーザー ID が Developer for System z の使用を許可されていることが検証されます。

TCP/IP が使用する SSL セキュリティー区分に詳しくれば、上記の検証ステップの組み合わせが「レベル 3 クライアント認証」仕様 (使用可能な最高レベル) に相当することがわかるはずです。

認証局 (CA) の妥当性検査

証明書の妥当性検査の一環として、証明書にユーザーの信頼する証明局 (CA) の署名があるかどうかを検査されます。それを行うために、RSE デーモンは CA を識別する証明書にアクセスできなければなりません。

SSL 接続に **gskkyman** 鍵データベースを使用している場合は、鍵データベースに CA 証明書を追加する必要があります。

SAF 鍵リングを使用している場合は (これが推奨される方式です)、次のサンプル RACF コマンドに示すように、CA 証明書をセキュリティー・データベースに CERTAUTH 証明書として、TRUST または HIGHTRUST 属性付きで追加する必要があります。

- RACDCERT CERTAUTH ADD(dsn) HIGHTRUST WITHLABEL('label')

ほとんどのセキュリティー製品では、すでにそのデータベースの中に、既知の CA の証明書が NOTRUST 状況付きで入っていることに注意してください。既存の CA 証明書をリストし、割り当てられたラベルに基づいて、その 1 つに信頼できる証明書としてマークを付けるには、次のサンプルの RACF コマンドを使用します。

- RACDCERT CERTAUTH LIST
- RACDCERT CERTAUTH ALTER(LABEL('HighTrust CA')) HIGHTRUST

注: 証明書内の HostIdMappings 拡張に基づいて RACF でユーザーを認証する場合は、HIGHTRUST 状況が必要です。詳細については、184 ページの『使用しているセキュリティー・ソフトウェアによる認証』を参照してください。

CA 証明書をセキュリティー・データベースに追加した後、このサンプルの RACF コマンドに示すように、CA 証明書を RSE 鍵リングに接続する必要があります。

- RACDCERT ID(stcrse) CONNECT(CERTAUTH LABEL('HighTrust CA')
RING(rdzssl.racf))

RACDCERT コマンドの詳細については、「*Security Server RACF コマンド言語解説書*」(SA88-8617) を参照してください。

重要: セキュリティー・ソフトウェアではなく RSE デーモンでユーザーを認証する場合は、SAF 鍵リングまたは **gskkyman** 鍵データベース内で TRUST 状況と HIGHTRUST 状況の CA が混在しないように注意する必要があります。RSE デーモンは、この 2 つを区別できないので、TRUST 状況の CA によって署名された証明書がユーザー ID 認証用に有効になります。

(オプション) 証明書失効リスト (CRL) に対する照会

必要であれば RSE デーモンに指示して、1 つ以上の証明書失効リスト (CRL) を検査させ、妥当性検査プロセスに追加のセキュリティーを付加できます。これは、CRL に関連した環境変数を `rsed.envvars` に追加することによって行います。以下のサンプルの変数については、35 ページの『`rsed.envvars`、RSE 構成ファイル』を参照してください。

- `GSK_CRL_SECURITY_LEVEL`
- `GSK_LDAP_SERVER`
- `GSK_LDAP_PORT`
- `GSK_LDAP_USER`
- `GSK_LDAP_PASSWORD`

これらの環境変数、および z/OS System SSL によって使用されるその他の環境変数の詳細については、「*Cryptographic Services System SSL (Secure Sockets Layer) プログラミング*」(SD88-6252) を参照してください。

注: 他の z/OS System SSL 環境変数 (`GSK_*`) を `rsed.envvars` で指定するときは、それらの環境変数によって、RSE デーモンによる SSL 接続と証明書認証の処理方法が変更される可能性があるので注意してください。

使用しているセキュリティー・ソフトウェアによる認証

RACF は、いくつかの検査を行って証明書を認証し、関連するユーザー ID を返します。他のセキュリティー製品では、これが異なる方法で行われる場合があることに注意してください。認証 (照会モード) を行うために使用される `initACEE` 機能の詳細については、使用しているセキュリティー製品の資料を参照してください。

1. RACF は、証明書が `DIGTCERT` クラス内で定義されているかどうかを検査します。定義されている場合、RACF は、その証明書が RACF データベースに追加されたときにその証明書に関連付けられていたユーザー ID を返します。

証明書を RACF に対して定義するには、次の例に示すような `RACDCERT` コマンドを使用します。

```
RACDCERT ID(userid) ADD(dsn) TRUST WITHLABEL('label')
```

2. 証明書が定義されていない場合、RACF は `DIGTNMAP` クラスまたは `DIGTCRIT` クラス内に一致する証明書名フィルターが定義されているかどうかを調べます。定義されていれば、最もよく一致するフィルターに関連したユーザー ID を返します。

注: Developer for System z が使用する証明書には、名前フィルターを使用しないことをお勧めします。それらのフィルターによって、すべての証明書が単

一のユーザー ID にマップされてしまうからです。その結果、すべての Developer for System z ユーザーが同じユーザー ID でログオンすることになります。

3. 一致する名前フィルターがない場合、RACF は HostIdMappings 証明書拡張を見つけ、埋め込まれたユーザー ID とホスト名のペアを抽出します。検出され、妥当性が確認された場合、RACF は HostIdMappings 拡張の中で定義されているユーザー ID を返します。

ユーザー ID とホスト名のペアは、以下のすべての条件が真である場合に有効です。

- この証明書に署名するために使用された CA 証明書に、DIGTCERT クラス内で HIGHTRUST のマークが付いている。
- 拡張内に保管されているユーザー ID の長さが妥当 (1 から 8 文字まで) である。
- RSE デーモンへ割り当てられたユーザー ID が、SERVAUTH クラス内の IRR.HOST.hostname プロファイルに対して (最低でも) READ 権限を持っている。ここで、hostname は拡張内に保管されているホスト名です。これは通常、CDFMVS08.RALEIGH.IBM.COM などのドメイン・ネームです。

ASN.1 構文での HostIdMappings 拡張の定義は、以下のとおりです。

```
id-ce-hostIdMappings OBJECT IDENTIFIER ::= { 1 3 18 0 2 18 1 }
HostIdMappings ::= SET OF HostIdMapping
HostIdMapping ::= SEQUENCE {
    hostName          IMPLICIT[1] IA5String,
    subjectId         IMPLICIT[2] IA5String,
    proofOfIdPossession IdProof OPTIONAL
}
IdProof ::= SEQUENCE {
    secret            OCTET STRING,
    encryptionAlgorithm OBJECT IDENTIFIER
}
```

注: HostIdMappings 拡張は、ターゲット・ユーザー ID が、その HostIdMappings 拡張を含んでいる証明書の有効期間の開始後に作成されている場合、受け入れられません。したがって、特に HostIdMappings 拡張を持つ証明書用にユーザー ID を作成する場合は、必ず、証明書要求を実行依頼する前にユーザー ID を作成しておいてください。

X.509 証明書、RACF によるその管理方法、および証明書名フィルターの定義方法について詳しくは、「*Security Server RACF セキュリティー管理者のガイド*」(SA88-8613)を参照してください。**RACDCERT** コマンドの詳細については、「*Security Server RACF コマンド言語解説書*」(SA88-8617)を参照してください。

RSE デーモンによる認証

Developer for System z では、ご使用のセキュリティー製品に依存せずに基本的な X.509 証明書認証を行うことができます。RSE デーモンによって行われる認証では、ユーザー ID とホスト名が証明書拡張内で定義されている必要があり、しかも、rsed.envvars 内の enable.certificate.mapping ディレクティブが FALSE に設定されている場合にのみ、認証がアクティブになります。

この機能は、使用しているセキュリティー製品が X.509 証明書に基づいたユーザーの認証をサポートしていない場合、または、使用しているセキュリティー製品によって行われるテストに証明書が合格できない場合 (例えば、証明書の HostIdMappings 拡張の ID に誤りがあり、DIGTCERT 内に名前フィルターや定義がない場合) に使用するためのものです。

クライアントはユーザーに対し、使用すべき拡張 ID (OID) を照会し、その ID は、デフォルトでは HostIdMappings OID の {1 3 18 0 2 18 1} です。

RSE デーモンは HostIdMappings 拡張のフォーマットを使用して、そこからユーザー ID とホスト名を抽出します。このフォーマットについては、184 ページの『使用しているセキュリティー・ソフトウェアによる認証』に説明があります。

ユーザー ID とホスト名のペアは、以下のすべての条件が真である場合に有効です。

- 拡張内に保管されているユーザー ID の長さが妥当 (1 から 8 文字まで) である。
- RSE デーモンへ割り当てられたユーザー ID が、SERVAUTH クラス内の IRR.HOST.hostname プロファイルに対して (最低でも) READ 権限を持っている。ここで、hostname は拡張内に保管されているホスト名です。これは通常、CDFMVS08.RALEIGH.IBM.COM などのドメイン・ネームです。

重要: RSE デーモンに既知のすべての CA を「高度に信頼できる」ものと保証するのは、セキュリティー管理者の責任です。RSE デーモンはクライアント証明書の署名者が「高度に信頼できる」か単なる「信頼できる」かを検査できないからです。アクセス可能な CA 証明書の詳細については、183 ページの『認証局 (CA) の妥当性検査』を参照してください。

Port Of Entry (POE) 検査

Developer for System z は Port Of Entry (POE) 検査をサポートしています。これにより、ホストは信頼できる TCP/IP アドレスにのみアクセスできます。このフィーチャーは、デフォルトでは使用不可に設定され、以下のサンプル RACF コマンドに示すように、BPX.POE セキュリティー・プロファイルの定義を必要とします。

- RDEFINE FACILITY BPX.POE UACC(NONE)
- PERMIT BPX.POE CLASS(FACILITY) ACCESS(READ) ID(STCRSE)
- SETROPTS RACLIST(FACILITY) REFRESH

注:

- 46 ページの『_RSE_JAVAOPTS での追加 Java 始動パラメーターの定義』の説明にあるように、rsed.envvars で「enable.port.of.entry=true」オプションをコメント解除することにより、POE を使用するよう RSE を構成する必要があります。
- RSE ユーザー ID STCRSE は、このプロファイルが定義されておらず、rsed.envvars 内で POE 検査が使用可能にされている場合、UID(0) を必要とします。
- BPX.POE を定義すると、POE 検査をサポートしている他の TCP/IP アプリケーション (INETD など) に影響が及びます。

- POE 検査の強度を完全に活用するには、SERVAUTH クラス内でセキュリティー・ゾーン (IP アドレス範囲である EZB.NETACCESS.** プロファイル) をセットアップしてください。

POE 検査を使用したネットワーク・アクセス制御の詳細については、「*Communications Server IP 構成ガイド*」(SC88-8926) を参照してください。

CICSTS セキュリティー

Developer for System z では、Application Deployment Manager を通じて、開発者が編集可能な CICS リソース定義とそのデフォルト値、および CICS リソース定義の表示を、CICS 管理者が CICS リソース定義 (CRD) サーバーによって制御できます。必要な CICS TS セキュリティー定義の詳細については、269 ページの『第 15 章 CICSTS に関する考慮事項』を参照してください。

CRD リポジトリ

CRD サーバー・リポジトリ VSAM データ・セットは、すべてのデフォルト・リソース定義を保持しています。したがって、更新されないように保護する必要がありますが、開発者は、そこに保管された値の読み取りを許可される必要があります。

CICS トランザクション

Developer for System z は、CICS リソースの定義および照会時に、CRD サーバーが使用する複数のトランザクションを提供します。トランザクションが接続すると、CICS リソース・セキュリティー検査機能 (使用可能な場合) により、ユーザー ID にはそのトランザクション ID を実行する許可が与えられます。

SSL 暗号化通信

Application Deployment Manager クライアントは、CICS TS Web サービスまたは RESTful インターフェースを使用して、CRD サーバーを起動します。この通信への SSL の使用は、CICS TS TCPIPSERVICE 定義によって制御されます。これについては、「*RACF Security Guide for CICS TS*」を参照してください。

SCLM セキュリティー

SCLM Developer Toolkit サービスは、ビルド、プロモート、およびデプロイ機能に対するオプションのセキュリティー機能を提供します。

SCLM 管理者による機能に対してセキュリティーが有効の場合、保護された機能呼び出し元ユーザー ID または代理ユーザー ID で実行する権限を確認するために、SAF 呼び出しが行われます。

必要な SCLM セキュリティー定義の詳細については、「*SCLM Developer Toolkit 管理者ガイド*」(SC88-5664) を参照してください。

Developer for System z 構成ファイル

セキュリティーのセットアップに影響するディレクティブは、Developer for System z の複数の構成ファイルに存在します。この章の情報に基づいて、セキュリティー管理者およびシステム・プログラマーは、以下のディレクティブの設定内容を決定することができます。

JES ジョブ・モニター - FEJJCNFG

- `LIMIT_COMMANDS={USERID | LIMITED | NOLIMIT}`

どのジョブに対してアクションを実行できるかを定義します (ブラウズと実行依頼は除く)。詳しくは、177 ページの『ジョブに対するアクション - ターゲットの制限』を参照してください。

- `LIMIT_VIEW={USERID | NOLIMIT}`

どのスプール・ファイルをブラウズできるかを定義します。詳しくは、180 ページの『スプール・ファイルへのアクセス』を参照してください。

- `APPLID={FEKAPPL | *}`

PassTicket の作成と妥当性検査に使用するアプリケーション ID。詳しくは、175 ページの『PassTicket の使用』を参照してください。

注: 上記の、およびその他の FEJJCNFG ディレクティブの詳細については、30 ページの『FEJJCNFG、JES ジョブ・モニター構成ファイル』に説明があります。

RSE - rsed.envvars

- `(_RSE_JAVAOPTS) -DDENY_PASSWORD_SAVE={true | false}`

ユーザーがホスト・パスワードをクライアント上に保存することを禁止します。詳しくは、46 ページの『_RSE_JAVAOPTS での追加 Java 始動パラメーターの定義』を参照してください。

- `(_RSE_JAVAOPTS) -DDSTORE_IDLE_SHUTDOWN_TIMEOUT=value`

活動停止中のクライアントを切断するタイマー。詳しくは、46 ページの『_RSE_JAVAOPTS での追加 Java 始動パラメーターの定義』を参照してください。

- `(_RSE_JAVAOPTS) -DAPPLID={FEKAPPL | *}`

PassTicket の作成と妥当性検査に使用するアプリケーション ID。詳しくは、175 ページの『PassTicket の使用』を参照してください。

- `(_RSE_JAVAOPTS) -Denable.port.of.entry={true | false}`

Port Of Entry 検査を使用可能にします。詳しくは、186 ページの『Port Of Entry (POE) 検査』を参照してください。

- `(_RSE_JAVAOPTS) -Denable.certificate.mapping={true | false}`

セキュリティー製品を使用して、X.509 証明書でユーザーを認証します。詳しくは、182 ページの『X.509 証明書を使用したクライアント認証』を参照してください。

- (`_RSE_JVAOPTS`) `-Ddaemon.log={/var/rdz/logs | *}`

監査ログ・ファイルのロケーション。詳しくは、176 ページの『監査ロギング』を参照してください。

注: 上記の、およびその他の `rsed.envvars` ディレクティブの詳細については、35 ページの『`rsed.envvars`、RSE 構成ファイル』に説明があります。

RSE - `ssl.properties`

- `daemon_keydb_file={SAF key ring name | gskkyman key database name}`

RSE デーモン証明書のロケーション。詳しくは、181 ページの『SSL 暗号化通信』を参照してください。

- `daemon_key_label=certificate label`

RSE デーモン証明書の名前。詳しくは、181 ページの『SSL 暗号化通信』を参照してください。

- `server_keystore_file={SAF key ring name | Java key store name}`

RSE サーバー証明書のロケーション。詳しくは、181 ページの『SSL 暗号化通信』を参照してください。

- `server_keystore_label=certificate label`

RSE サーバー証明書の名前。詳しくは、181 ページの『SSL 暗号化通信』を参照してください。

- `server_keystore_type={JKS | JCECARCFKS | JCECCARCFKS}`

使用される鍵ストアのタイプ (Java 鍵ストアまたは SAF 鍵リング)。詳しくは、181 ページの『SSL 暗号化通信』を参照してください。

注: 上記の、およびその他の `ssl.properties` ディレクティブの詳細については、100 ページの『(オプション) RSE SSL 暗号化』に説明があります。

セキュリティ定義

サンプル・メンバー `FEKRACF` をカスタマイズし、実行依頼してください。このメンバーには、Developer for System z 用の基本セキュリティ定義を作成する、サンプルの `RACF` および `z/OS UNIX` コマンドが含まれています。

`FEKRACF` は `FEK.#CUST.JCL` に置かれます。ただし、ジョブ `FEK.SFEKSAMP` (`FEKSETUP`) をカスタマイズして実行依頼したときに別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

`RACF` コマンドの詳細については、「*RACF コマンド言語解説書*」(SA88-8617) を参照してください。

注:

- CA ACF2™ for z/OS を使用しているサイトの場合は、
<https://support.ca.com/irj/portal/kbtech?ipLogNrow=0&docid=492389>

&searchID=TEC492389 にアクセスして、Developer for System z を正しく構成するために必要なセキュリティー・コマンドの詳細を参照してください。

- CA Top Secret® for z/OS を使用しているサイトの場合は、CA サポート・サイト (<https://support.ca.com>) で、ご使用の製品のページを参照し、関連する Developer for System z Knowledge Document を確認してください。この Knowledge Document には、Developer for System z を正しく構成するために必要なセキュリティー・コマンドの詳細が記載されています。

以下のセクションでは、必要なステップ、オプションの構成、および可能な代替策について説明します。

要件およびチェックリスト

セキュリティー・セットアップを完了するために、セキュリティー管理者は表 26 にリストされた値を認識しておく必要があります。これらの値は、前のステップである Developer for System z のインストールとカスタマイズで定義されています。

表 26. セキュリティー・セットアップの変動要素

説明	<ul style="list-style-type: none">デフォルト値正解の入手先	値
Developer for System z 製品 高位修飾子	<ul style="list-style-type: none">FEKSMP/E インストール	
Developer for System z カス タマイズ高位修飾子	<ul style="list-style-type: none">FEK.#CUSTFEK.SFEKSAMP(FEKSETUP) (18 ページの『カスタマイ ズのセットアップ』を参 照)	
JES ジョブ・モニター開始タ スク名	<ul style="list-style-type: none">JMONFEK.#CUST.PROCLIB(JMON) (24 ページの『PROCLIB の変更』を参照)	
RSE デーモン開始タスク名	<ul style="list-style-type: none">RSEDFEK.#CUST.PROCLIB(RSED) (24 ページの『PROCLIB の変更』を参照)	
ロック・デーモン開始タスク 名	<ul style="list-style-type: none">LOCKDFEK.#CUST.PROCLIB(LOCKD) (24 ページの『PROCLIB の変更』を参照)	
アプリケーション ID	<ul style="list-style-type: none">FEKAPPL/etc/rdz/rsed.envvars (46 ページの 『_RSE_JAVAOPTS での 追加 Java 始動パラメータ ーの定義』を参照)	

次のリストは、Developer for System z の基本的なセキュリティー・セットアップを完了するために必要なアクションの概要を示したものです。以下の各セクションで説明されているように、これらの要件を満たすために、求めるセキュリティー・レベルに応じてさまざまな方式を使用できます。オプションの Developer for System z サービスのセキュリティー・セットアップについては、前記の各セクションを参照してください。

- 『セキュリティーの設定およびクラスのアクティブ化』
- 192 ページの『Developer for System z ユーザーの OMVS セグメントの定義』
- 192 ページの『データ・セット・プロファイルの定義』
- 196 ページの『Developer for System z 開始タスクの定義』
- 197 ページの『JES コマンド・セキュリティーの定義』
- 199 ページの『セキュアな z/OS UNIX サーバーとしての RSE の定義』
- 199 ページの『RSE の MVS プログラム制御ライブラリーの定義』
- 200 ページの『RSE のアプリケーション保護の定義』
- 200 ページの『RSE サーバーの PassTicket サポートの定義』
- 201 ページの『RSE 用の z/OS UNIX プログラム制御ファイルの定義』
- 202 ページの『セキュリティー設定の検査』

セキュリティーの設定およびクラスのアクティブ化

Developer for System z では、さまざまなセキュリティー・メカニズムを使用して、クライアントにとってセキュアで制御されたホスト環境を確保します。そのためには、以下のサンプルの RACF コマンドで示すように、いくつかのクラスとセキュリティー設定をアクティブにする必要があります。

- 現行の設定を表示する
 - SETROPTS LIST
- z/OS UNIX およびデジタル証明書プロファイルのファシリティ・クラスをアクティブにする
 - SETROPTS GENERIC(FACILITY)
 - SETROPTS CLASSACT(FACILITY) RACLIST(FACILITY)
- 開始タスク定義をアクティブにする
 - SETROPTS GENERIC(STARTED)
 - RDEFINE STARTED ** STDATA(USER(=MEMBER) GROUP(STCGROUP) TRACE(YES))
 - SETROPTS CLASSACT(STARTED) RACLIST(STARTED)
- JES ジョブ・モニターのコンソール・セキュリティーをアクティブにする
 - SETROPTS GENERIC(CONSOLE)
 - SETROPTS CLASSACT(CONSOLE) RACLIST(CONSOLE)
- JES ジョブ・モニターのオペレーター・コマンド保護をアクティブにする
 - SETROPTS GENERIC(OPERCMDS)
 - SETROPTS CLASSACT(OPERCMDS) RACLIST(OPERCMDS)
- RSE のアプリケーション保護をアクティブにする
 - SETROPTS GENERIC(APPL)
 - SETROPTS CLASSACT(APPL) RACLIST(APPL)
- RSE の PassTicket を使用したセキュアなサインオンをアクティブにする
 - SETROPTS GENERIC(PTKTDATA)

- SETROPTS CLASSACT(PTKTDATA) RACLIST(PTKTDATA)
- 信頼されたコードだけを RSE がロードできるように、プログラム制御をアクティブにする
 - RDEFINE PROGRAM ** ADDMEM('SYS1.CMDLIB'//NOPADCHK) UACC(READ)
 - SETROPTS WHEN(PROGRAM)

注: すでに PROGRAM クラス内に * プロファイルがある場合は、** プロファイルを作成しないでください。セキュリティ・ソフトウェアによって使用される検索パスが、分かりにくく、複雑なものになります。その場合は、既存の * 定義と新しい ** 定義をマージする必要があります。IBM では、** プロファイルの使用を推奨しています。これについては、「*Security Server RACF セキュリティ管理者のガイド*」(SA88-8613) に説明があります。

重要: 「WHEN PROGRAM」がアクティブの場合、一部の製品 (FTP など) はプログラムで制御することが必要です。これは、実動システム上でアクティブにする前にテストしてください。

- (オプション) X.509 HostIdMappings および拡張 Port Of Entry (POE) サポートをアクティブにする
 - SETROPTS GENERIC(SERVAUTH)
 - SETROPTS CLASSACT(SERVAUTH) RACLIST(SERVAUTH)

Developer for System z ユーザーの OMVS セグメントの定義

Developer for System z のユーザーごとに、有効なゼロ以外の z/OS UNIX ユーザー ID (UID)、ホーム・ディレクトリー、およびシェル・コマンドを指定する RACF OMVS セグメント (または同等のもの) を定義する必要があります。また、ユーザーのデフォルト・グループも、グループ ID を持つ OMVS セグメントを必要とします。

以下のサンプル RACF コマンドでは、#userid、#user-identifier、#group-name、#group-identifier の各プレースホルダーを実際の値に置き換えてください。

- ALTUSER #userid
OMVS(UID(#user-identifier) HOME(/u/#userid) PROGRAM(/bin/sh) NOASSIZEMAX)
- ALTGROUP #group-name OMVS(GID(#group-identifier))

お勧めはできませんが、FACILITY クラスの BPX.DEFAULT.USER プロファイルで定義されている共用 OMVS セグメントを使用して、Developer for System z の OMVS セグメント要件を満たすこともできます。

データ・セット・プロファイルの定義

ほとんどの Developer for System z データ・セットでは、ユーザーの場合は READ アクセス権、システム・プログラマーの場合は ALTER で十分です。#sysprog プレースホルダーは、有効なユーザー ID または RACF グループ名に置き換えてください。また、正しいデータ・セット名については、製品をインストールおよび構成したシステム・プログラマーに問い合わせてください。FEK は、インストール時に使用されたデフォルトの高位修飾子で、FEK.#CUST はカスタマイズ・プロセスで作成されたデータ・セットのデフォルトの高位修飾子です。

- ADDGROUP (FEK) OWNER(IBMUSER) SUPGROUP(SYS1)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z - HLQ STUB')
- ADDSD 'FEK.*.**' UACC(READ)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
- PERMIT 'FEK.*.**' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
- SETROPTS GENERIC(DATASET) REFRESH

注:

- FEK.SFEKAUTH は APF 許可があるデータ・セットであるため、更新されないように保護することを強くお勧めします。同じことは FEK.SFEKLOAD および FEK.SFEKLPA についても言えますが、ここでは、これらのデータ・セットがプログラムで制御されていることが理由です。
- この資料内および FEKRACF ジョブ内のサンプル・コマンドは、EGN (Enhanced Generic Naming) がアクティブであることを想定しています。これにより、** 修飾子を使用して、DATASET クラス内の任意の数の修飾子を表すことができます。使用しているシステムで EGN がアクティブでない場合は、** を * に置き換えてください。EGN の詳細については、「*Security Server RACF セキュリティー管理者のガイド*」(SA88-8613) を参照してください。

一部のオプションの Developer for System z コンポーネントには、追加のセキュリティー・データ・セット・プロファイルが必要です。#sysprog、#ram-developer、および #cicsadmin の各プレースホルダーは、有効なユーザー ID または RACF グループ名に置き換えてください。

- SCLM Developer Toolkit のロング/ショート・ネーム変換を使用している場合は、ユーザーにマッピング VSAM の FEK.#CUST.LSTRANS.FILE に対する UPDATE アクセス権が必要です。
 - ADDSD 'FEK.#CUST.LSTRANS.*.**' UACC(UPDATE)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z - SCLMDT')
 - PERMIT 'FEK.#CUST.LSTRANS.*.**' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
 - SETROPTS GENERIC(DATASET) REFRESH
- CARMA RAM (Repository Access Manager) 開発者には、CARMA VSAM である FEK.#CUST.CRA* に対する UPDATE アクセス権が必要です。
 - ADDSD 'FEK.#CUST.CRA*.*' UACC(READ)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z - CARMA')
 - PERMIT 'FEK.#CUST.CRA*.*' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
 - PERMIT 'FEK.#CUST.CRA*.*' CLASS(DATASET) ACCESS(UPDATE) ID(#ram-developer)
 - SETROPTS GENERIC(DATASET) REFRESH
- Application Deployment Manager の CRD サーバー (CICS リソース定義) を使用している場合は、CICS 管理者に、CRD リポジトリ VSAM に対する UPDATE アクセス権が必要です。
 - ADDSD 'FEK.#CUST.ADNREP*.*' UACC(READ)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z - ADN')
 - PERMIT 'FEK.#CUST.ADNREP*.*' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
 - PERMIT 'FEK.#CUST.ADNREP*.*' CLASS(DATASET) ACCESS(UPDATE) ID(#cicsadmin)
 - SETROPTS GENERIC(DATASET) REFRESH
- Application Deployment Manager のマニフェスト・リポジトリが定義されている場合は、すべての CICS Transaction Server ユーザーに、マニフェスト・リポジトリ VSAM に対する UPDATE アクセス権が必要です。

```

- ADDSD 'FEK.#CUST.ADNMAN*.*' UACC(UPDATE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z - ADN')
- PERMIT 'FEK.#CUST.ADNMAN*.*' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
- SETROPTS GENERIC(DATASET) REFRESH

```

READ アクセス権も制御対象とする、さらにセキュアなセットアップのためには、以下のサンプル RACF コマンドを使用します。

- uacc(none) データ・セット保護


```

- ADDGROUP (FEK)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z - HLQ STUB')
  OWNER(IBMUSER) SUPGROUP(SYS1)"
- ADDSD 'FEK*.*' UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
- ADDSD 'FEK.SFEKAUTH' UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
- ADDSD 'FEK.SFEKLOAD' UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
- ADDSD 'FEK.SFEKPROC' UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
- ADDSD 'FEK.#CUST.PARMLIB' UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
- ADDSD 'FEK.#CUST.CNTL' UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
- ADDSD 'FEK.#CUST.LSTRANS*.*' UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z - SCLMDT')
- ADDSD 'FEK.#CUST.CRA*.*' UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z - CARMA')
- ADDSD 'FEK.#CUST.ADNREP*.*' UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z - ADN')
- ADDSD 'FEK.#CUST.ADNMAN*.*' UACC(NONE)
  DATA('RATIONAL DEVELOPER FOR SYSTEM Z - ADN')

```
- システム・プログラマーにすべてのライブラリーの管理を許可する


```

- PERMIT 'FEK*.*' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
- PERMIT 'FEK.SFEKAUTH' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
- PERMIT 'FEK.SFEKLOAD' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
- PERMIT 'FEK.SFEKLOAD' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
- PERMIT 'FEK.SFEKLOAD' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
- PERMIT 'FEK.SFEKPROC' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
- PERMIT 'FEK.#CUST.PARMLIB' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
- PERMIT 'FEK.#CUST.CNTL' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
- PERMIT 'FEK.#CUST.LSTRANS*.*' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
- PERMIT 'FEK.#CUST.CRA*.*' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
- PERMIT 'FEK.#CUST.ADNREP*.*' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)
- PERMIT 'FEK.#CUST.ADNMAN*.*' CLASS(DATASET) ACCESS(ALTER) ID(#sysprog)

```
- クライアントにロードおよび exec ライブラリーへのアクセスを許可する


```

- PERMIT 'FEK.SFEKAUTH' CLASS(DATASET) ACCESS(READ) ID(*)
- PERMIT 'FEK.SFEKLOAD' CLASS(DATASET) ACCESS(READ) ID(*)
- PERMIT 'FEK.SFEKPROC' CLASS(DATASET) ACCESS(READ) ID(*)
- PERMIT 'FEK.#CUST.CNTL' CLASS(DATASET) ACCESS(READ) ID(*)

```

注: FEK.SFEKLPA に許可は必要ありません。LPA 内に存在するすべてのコードには、全員がアクセスできるからです。

- JES ジョブ・モニターにロードおよびパラメーター・ライブラリーへのアクセスを許可する
 - PERMIT 'FEK.SFEKAUTH' CLASS(DATASET) ACCESS(READ) ID(STCJMON)
 - PERMIT 'FEK.#CUST.PARMLIB' CLASS(DATASET) ACCESS(READ) ID(STCJMON)
- (オプション) クライアントに SCLMDT 用のロング/ショート・ネーム変換 VSAM 更新を許可する
 - PERMIT 'FEK.#CUST.LSTRANS*.**' CLASS(DATASET) ACCESS(UPDATE) ID(*)
- (オプション) RAM 開発者に CARMA 用の CARMA VSAM の更新を許可する
 - PERMIT 'FEK.#CUST.CRA*.**' CLASS(DATASET) ACCESS(UPDATE) ID(#ram-developer)
- (オプション) CICS ユーザーに、Application Deployment Manager 用の CRD リポジトリ VSAM の読み取りを許可します。
 - PERMIT 'FEK.#CUST.ADNREP*.**' CLASS(DATASET) ACCESS(READ) ID(*)
- (オプション) CICS 管理者に、Application Deployment Manager 用の CRD リポジトリ VSAM の更新を許可します。
 - PERMIT 'FEK.#CUST.ADNREP*.**' CLASS(DATASET) ACCESS(UPDATE) ID(#cicsadmin)
- (オプション) CICS ユーザーに、Application Deployment Manager 用のマニフェスト・リポジトリ VSAM の更新を許可します。
 - PERMIT 'FEK.#CUST.ADNMAN*.**' CLASS(DATASET) ACCESS(UPDATE) ID(*)
- (オプション) CICS TS サーバーに bidi 用のロード・ライブラリーと Application Deployment Manager へのアクセスを許可する
 - PERMIT 'FEK.SFEKLOAD' CLASS(DATASET) ACCESS(READ) ID(#cicsts)
- (オプション) DB2 サーバーに DB2 ストアド・プロシージャ・ビルダーの exec ライブラリーへのアクセスを許可する
 - PERMIT 'FEK.SFEKPROC' CLASS(DATASET) ACCESS(READ) ID(#db2)
- セキュリティー・プロファイルをアクティブにする
 - SETROPTS GENERIC(DATASET) REFRESH

システム・データ・セットへの READ アクセスを制御するときは、Developer for System z サーバーおよびユーザーに以下のデータ・セットに対する READ 権限を与える必要があります。

- CEE.SCEERUN
- CEE.SCEERUN2
- CBC.SCLBDLL
- ISP.SISPLD
- ISP.SISPLPA
- SYS1.LINKLIB
- SYS1.SIEALNKE
- REXX.V1R4M0.SEAGLPA

注: REXX 製品パッケージに代替ライブラリーを使用している場合、デフォルトの REXX ランタイム・ライブラリー名は REXX*.SEAGALT です。上記の例で使われているような REXX*.SEAGLPA ではありません。

Developer for System z 開始タスクの定義

以下のサンプル RACF コマンドは、保護されたユーザー ID (STCJMON、STCRSE、および STCLOCK) とそれらに割り当てられたグループ STCGROUP を使用して、JMON、RSED、および LOCKD の各開始タスクを作成します。#group-id および #user-id-* プレースホルダーは、有効な OMVS ID に置き換えてください。

- ADDGROUP STCGROUP OMVS(GID(#group-id))
DATA('GROUP WITH OMVS SEGMENT FOR STARTED TASKS')
- ADDUSER STCJMON DFLTGROUP(STCGROUP) NOPASSWORD NAME('RDZ - JES JOBMONITOR')
OMVS(UID(#user-id-jmon) HOME(/tmp) PROGRAM(/bin/sh) NOASSIZEMAX
NOTHREADSMAX)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
- ADDUSER STCRSE DFLTGROUP(STCGROUP) NOPASSWORD NAME('RDZ - RSE DAEMON')
OMVS(UID(#user-id-rse) HOME(/tmp) PROGRAM(/bin/sh) ASSIZEMAX(2147483647)
NOTHREADSMAX)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
- ADDUSER STCLOCK DFLTGROUP(STCGROUP) NOPASSWORD NAME('RDZ - LOCK DAEMON')
OMVS(UID(#user-id-lock) HOME(/tmp) PROGRAM(/bin/sh) NOASSIZEMAX)
NOTHREADSMAX)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
- RDEFINE STARTED JMON.* DATA('RDZ - JES JOBMONITOR')
STDATA(USER(STCJMON) GROUP(STCGROUP) TRUSTED(NO))
- RDEFINE STARTED RSED.* DATA('RDZ - RSE DAEMON')
STDATA(USER(STCRSE) GROUP(STCGROUP) TRUSTED(NO))
- RDEFINE STARTED LOCKD.* DATA('RDZ - LOCK DAEMON')
STDATA(USER(STCLOCK) GROUP(STCGROUP) TRUSTED(NO))
- SETROPTS RACLIST(STARTED) REFRESH

注:

1. NOPASSWORD キーワードを指定することにより、開始タスクのユーザー ID が必ず保護されるようにしてください。
2. RSE サーバーが固有の OMVS uid を持つようにしてください (その uid へ付与される z/OS UNIX 関連の特権のため)。
3. RSE デーモンを適切に動作させるためには、大きなサイズのアドレス・スペース (2 GB) が必要です。この値は、ユーザー ID STCRSE の OMVS セグメントの ASSIZEMAX 変数で設定してください。これにより、SYS1.PARMLIB(BPXPRMxx) で MAXASSIZE が変更されても、RSE デーモンには必要な領域サイズが確実に設定されます。
4. RSE を適切に動作させるためには、多数のスレッドも必要です。スレッド数の限度は、ユーザー ID STCRSE の OMVS セグメントの THREADSMAX 変数で設定できます。これにより、SYS1.PARMLIB(BPXPRMxx) で MAXTHREADS または MAXTHREADTASKS が変更されても、RSE には必要なスレッド数の限度が確実に設定されます。スレッド数の限度の適切な値を特定するには、227 ページの『第 13 章 チューニングに関する考慮事項』を参照してください。
5. ユーザー ID STCJMON も、OMVS セグメントで THREADSMAX を設定する対象となります。これは、JES ジョブ・モニターがクライアント接続ごとにスレッドを使用するためです。

STCRSE ユーザー ID の制限を考慮することが必要になる場合があります。

RESTRICTED 属性を持つユーザーは、保護された (MVS) リソースには、特にアクセスを許可されている場合以外、アクセスできません。

ALTUSER STCRSE RESTRICTED

制限されたユーザーが「その他の」許可ビットによって z/OS UNIX ファイル・システム・リソースにアクセスできないよう、UNIXPRIV クラス内に RESTRICTED.FILESYS.ACCESS プロファイルを UACC(NONE) で定義する必要があります。ユーザー ID を制限する方法の詳細については、「*Security Server RACF セキュリティー管理者のガイド*」(SA88-8613) を参照してください。

重要: 制限されたユーザー ID を使用している場合は、TSO の **PERMIT** コマンドか、z/OS UNIX **setfacl** コマンドを使用して、リソースにアクセスする許可を明示的に追加する必要があります。これには、Developer for System z 資料が UACC を使用するリソース (PROGRAM クラス内の ** プロファイルなど)、またはそれが共通の z/OS UNIX 規則 (全員が Java ライブラリーの読み取り権限と実行権限を持つなど) に依存するリソースが含まれます。これは、実動システム上でアクティブにする前にテストしてください。

JES コマンド・セキュリティの定義

JES ジョブ・モニターは、ユーザーが要求したすべての JES オペレーター・コマンドを、拡張 MCS (EMCS) コンソールを通じて発行します。このコンソールの名前は、30 ページの『FEJCNFG、JES ジョブ・モニター構成ファイル』の説明にあるように、CONSOLE_NAME ディレクティブによって制御されます。

以下のサンプル RACF コマンドは、Developer for System z ユーザーに、JES コマンドの限定セット (保留、保留解除、キャンセル、およびパージ) に対する条件付きアクセス権を与えます。ユーザーは、JES ジョブ・モニターによってコマンドを発行した場合にのみ、実行権限を持ちます。#console プレースホルダーは、実際のコンソール名に置き換えてください。

- RDEFINE OPERCMDS MVS.MCSOPER.#console UACC(READ)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
- RDEFINE OPERCMDS JES%.** UACC(NONE)
- PERMIT JES%.** CLASS(OPERCMDS) ACCESS(UPDATE) WHEN(CONSOLE(JMON)) ID(*)
- SETROPTS RACLIST(OPERCMDS) REFRESH

注:

- コンソールの使用は、MVS.MCSOPER.#console プロファイルが定義されていない場合に許可されます。
- WHEN(CONSOLE(JMON)) が機能するためには、CONSOLE クラスがアクティブでなければなりませんが、CONSOLE クラス内に EMCS コンソールがあるかどうかについての実際のプロファイル検査はありません。
- WHEN(CONSOLE(JMON)) 文節内で、JMON を実際のコンソール名に置き換えないでください。JMON キーワードは、コンソール名ではなく、入り口点アプリケーションを表しています。

重要: ご使用のセキュリティ・ソフトウェアで汎用アクセス NONE を使用して JES コマンドを定義すると、他のアプリケーションや操作に影響が出る場合があります。これは、実動システム上でアクティブにする前にテストしてください。

198 ページの表 27 および 198 ページの表 28 は、JES2 および JES3 について発行されたオペレーター・コマンドと、それらを保護するために使用できる個別セキュリティ・プロファイルを示しています。

表 27. JES2 ジョブ・モニターのオペレーター・コマンド

アクション	コマンド	OPERCMDS プロファイル	必要なアクセス権
保留	\$Hx(jobid) x = {J、S、または T}	jesname.MODIFYHOLD.BAT jesname.MODIFYHOLD.STC jesname.MODIFYHOLD.TSU	UPDATE
保留解除	\$Ax(jobid) x = {J、S、または T}	jesname.MODIFYRELEASE.BAT jesname.MODIFYRELEASE.STC jesname.MODIFYRELEASE.TSU	UPDATE
キャンセル	\$Cx(jobid) x = {J、S、または T}	jesname.CANCEL.BAT jesname.CANCEL.STC jesname.CANCEL.TSU	UPDATE
パージ	\$Cx(jobid),P x = {J、S、または T}	jesname.CANCEL.BAT jesname.CANCEL.STC jesname.CANCEL.TSU	UPDATE

表 28. JES3 ジョブ・モニターのオペレーター・コマンド

アクション	コマンド	OPERCMDS プロファイル	必要なアクセス権
保留	*F,J=jobid,H	jesname.MODIFY.JOB	UPDATE
保留解除	*F,J=jobid,R	jesname.MODIFY.JOB	UPDATE
キャンセル	*F,J=jobid,C	jesname.MODIFY.JOB	UPDATE
パージ	*F,J=jobid,C	jesname.MODIFY.JOB	UPDATE

注:

- 「保留」、「保留解除」、「キャンセル」、「パージ」の各 JES オペレーター・コマンドと「JCL の表示」コマンドは、クライアント・ユーザー ID が所有しているスプール・ファイルに対してのみ実行できます。ただし、JES ジョブ・モニター構成ファイル内で、LIMIT_COMMANDS= が値 LIMITED または NOLIMIT に指定されている場合は除きます。それについての詳細は、177 ページの『ジョブに対するアクション - ターゲットの制限』を参照してください。
- ユーザーは、JES ジョブ・モニター構成ファイル内で LIMIT_VIEW=USERID が定義されている場合を除き、すべてのスプール・ファイルを参照できます。それについての詳細は、180 ページの『スプール・ファイルへのアクセス』を参照してください。
- これらのオペレーター・コマンドの許可がない場合でも、ユーザーは、これらのリソースを保護できるプロファイル (JESINPUT、JESJOBS、および JESSPOOL クラス内のプロファイルなど) に対して十分な権限を持っていれば、JES ジョブ・モニターを通じてジョブを実行依頼し、ジョブ出力を読み取ることができます。

TSO セッションから JMON コンソールを作成することによって JES ジョブ・モニター・サーバーの ID を装うことは、セキュリティ・ソフトウェアによって防止されます。コンソールを作成できる場合でも、(JES ジョブ・モニターと TSO とでは) 入り口点が異なります。この資料で説明されているとおりにセキュリティが

セットアップされており、ユーザーが他の手段によって JES コマンドに対する権限を持っていない場合は、そのコンソールから発行された JES コマンドはセキュリティ検査で不合格になります。

セキュアな z/OS UNIX サーバーとしての RSE の定義

RSE は、クライアントのスレッドのセキュリティ環境を作成または削除するためには、BPX.SERVER プロファイルに対する UPDATE アクセス権を必要とします。このプロファイルが定義されていない場合は、UID(0) が RSE に必要です。

- RDEFINE FACILITY BPX.SERVER UACC(NONE)
- PERMIT BPX.SERVER CLASS(FACILITY) ACCESS(UPDATE) ID(STCRSE)
- SETROPTS RACLIST(FACILITY) REFRESH

重要: BPX.SERVER プロファイルを定義すると、z/OS UNIX 全体が UNIX レベルのセキュリティから、より安全な z/OS UNIX レベルのセキュリティに切り替わります。これによって、他の z/OS UNIX アプリケーションと操作が影響を受ける場合もあります。これは、実動システム上でアクティブにする前にテストしてください。さまざまなセキュリティ・レベルの詳細については、「UNIX System Services 計画」(GA88-8639) を参照してください。

RSE の MVS プログラム制御ライブラリーの定義

BPX.SERVER に対する権限を持つサーバーは、クリーンなプログラム制御環境で実行する必要があります。これは、RSE によって呼び出されるすべてのプログラムも、プログラムで制御する必要があることを意味します。MVS ロード・ライブラリーの場合、プログラム制御はセキュリティ・ソフトウェアによって管理されます。

RSE は、システム (SYS1.LINKLIB)、言語環境プログラムのランタイム (CEE.SCEERUN*)、および ISPF の TSO/ISPF クライアント・ゲートウェイ (ISP.SISPLOAD) ロード・ライブラリーを使用します。

- RALTER PROGRAM ** UACC(READ) ADDMEM('SYS1.LINKLIB'//NOPADCHK)
- RALTER PROGRAM ** UACC(READ) ADDMEM('CEE.SCEERUN'//NOPADCHK)
- RALTER PROGRAM ** UACC(READ) ADDMEM('CEE.SCEERUN2'//NOPADCHK)
- RALTER PROGRAM ** UACC(READ) ADDMEM('ISP.SISPLOAD'//NOPADCHK)
- SETROPTS WHEN(PROGRAM) REFRESH

注: すでに PROGRAM クラス内に * プロファイルがある場合は、** プロファイルを使用しないでください。セキュリティ・ソフトウェアによって使用される検索パスが、分かりにくく、複雑なものになります。その場合は、既存の * 定義と新しい ** 定義をマージする必要があります。IBM では、** プロファイルの使用を推奨しています。これについては、「Security Server RACF セキュリティ管理者のガイド」(SA88-8613) に説明があります。

オプションのサービスを使用できるようにするには、以下の (前提条件の) 追加ライブラリーがプログラムで制御されるようにする必要があります。このリストには、Developer for System z が対話する製品 (IBM Debug Tool など) に固有のデータ・セットは含まれていません。

- 代替 REXX ランタイム・ライブラリー (SCLM Developer Toolkit 用)

- REXX.*.SEAGALT
- システム・ロード・ライブラリー (SSL 暗号化用)
 - SYS1.SIEALNKE
- File Manager リスナー・ロード・ライブラリー (File Manager Integration 用)
 - FMN.SFMNMODA

注: LPA 配置用に設計されたライブラリーは、LINKLIST または STEPLIB によってアクセスされる場合、追加のプログラム制御権限も必要とします。この資料では、以下の LPA ライブラリーの使用法について説明します。

- ISPF (TSO/ISPF クライアント・ゲートウェイ用)
 - ISP.SISPLPA
- REXX ランタイム・ライブラリー (SCLM Developer Toolkit 用)
 - REXX.*.SEAGLPA
- Developer for System z (CARMA 用)
 - FEK.SFEKLPA

RSE のアプリケーション保護の定義

クライアントがログオンするときに、RSE デーモンはユーザーがアプリケーションの使用を許可されていることを検証します。

- RDEFINE APPL FEKAPPL UACC(READ) DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
- SETROPTS RACLIST(APPL) REFRESH

注:

『RSE サーバーの PassTicket サポートの定義』で詳しく説明するように、RSE は FEKAPPL 以外のアプリケーション ID の使用をサポートしています。APPL クラス定義は、RSE が使用する実際のアプリケーション ID と一致している必要があります。

重要: アプリケーション・プロファイルが定義されていない場合、またはユーザーにプロファイルへの READ アクセス権がない場合は、クライアントの接続要求が失敗します。

RSE サーバーの PassTicket サポートの定義

クライアントのパスワード (または、X.509 証明書などのその他の識別手段) は、接続時にクライアントの ID を検査するためにのみ使用されます。その後は、スレッド・セキュリティを維持するために PassTicket が使用されます。

PassTicket は、有効期間が約 10 分のシステム生成パスワードです。生成される PassTicket は、秘密鍵に基づいています。この鍵は、64 ビットの数値 (16 個の 16 進文字) です。以下のサンプル RACF コマンドでは、key16 プレースホルダーをユーザー指定の 16 文字の 16 進ストリング (0 から 9 までと A から F までの文字) に置き換えてください。

- RDEFINE PTKTDATA FEKAPPL UACC(NONE) SSIGNON(KEYMASKED(key16))
 APPLDATA('NO REPLAY PROTECTION - DO NOT CHANGE')
 DATA('RATIONAL DEVELOPER FOR SYSTEM Z')

- RDEFINE PTKTDATA IRRPTAUTH.FEKAPPL.* UACC(NONE)
DATA('RATIONAL DEVELOPER FOR SYSTEM Z')
- PERMIT IRRPTAUTH.FEKAPPL.* CLASS(PTKTDATA) ACCESS(UPDATE) ID(STCRSE)
- SETROPTS RACLIST(PTKTDATA) REFRESH

RSE は、FEKAPPL 以外のアプリケーション ID の使用をサポートしています。これをアクティブにするには、46 ページの『_RSE_JAVAOPTS での追加 Java 始動パラメーターの定義』で説明しているように、rsed.envvars 内で「APPLID=FEKAPPL」オプションをコメント解除およびカスタマイズします。PTKTDATA クラス定義は、RSE が使用する実際のアプリケーション ID と一致している必要があります。

OMVSAPPL はアプリケーション ID として使用しないでください。これは、大部分の z/OS UNIX アプリケーションの秘密鍵を公開するからです。また、デフォルトの MVS アプリケーション ID (MVS の直後にシステムの SMF ID を続けたもの) も使用しないでください。これは、大部分の MVS アプリケーション (ユーザー・バッチ・ジョブを含む) の秘密鍵を公開するからです。

注:

- PTKTDATA クラスがすでに定義されている場合は、上記のリストにあるプロファイルを作成する前に、それが総称クラスとして定義されていることを確認してください。PTKTDATA クラス内の総称文字のサポートは、PassTicket に Java インターフェースが導入された z/OS リリース 1.7 からの新機能です。
- RSE が PassTicket を生成できるユーザー ID を制限するには、IRRPTAUTH.FEKAPPL.* 定義の中のワイルドカード (*) を、有効なユーザー ID マスクで置き換えます。
- RACF の設定によっては、プロファイルを定義しているユーザーが、そのプロファイルのアクセス・リストにも入っている場合があります。PTKTDATA プロファイルについては、この許可を除去することをお勧めします。
- RSE が提示した PassTicket を JES ジョブ・モニターが評価できるようにするには、JES ジョブ・モニターと RSE が、同じアプリケーション ID を持っている必要があります。
- システムに暗号製品がインストールされており、使用可能になっている場合、保護されたサインオン・アプリケーション鍵を暗号化して、保護を強化することができます。それを行うには、KEYMASKED ではなく KEYENCRYPTED キーワードを使用します。これについて詳しくは、「*Security Server RACF セキュリティー管理者のガイド*」(SA88-8613) を参照してください。

重要: PassTicket が正しくセットアップされていないと、クライアントの接続要求は失敗します。

RSE 用の z/OS UNIX プログラム制御ファイルの定義

BPX.SERVER に対する権限を持つサーバーは、クリーンなプログラム制御環境で実行する必要があります。これは、RSE によって呼び出されるすべてのプログラムも、プログラムで制御する必要があることを意味します。z/OS UNIX ファイルの場合、プログラム制御は **extattr** コマンドによって管理されます。このコマンドを実行するには、FACILITY クラス内の BPX.FILEATTR.PROGCTL に対する READ アクセス権を持つか、または UID(0) であることが必要です。

RSE サーバーは、RACF の Java 共用ライブラリー (/usr/lib/libIRRacsf.so) を使用します。

- extattr +p /usr/lib/libIRRacsf.so

注:

- z/OS 1.9 以降、/usr/lib/libIRRacsf.so は SMP/E RACF のインストール時にプログラムによる制御としてインストールされます。
- z/OS 1.10 以降、/usr/lib/libIRRacsf.so はベース z/OS に添付される SAF の一部であるので、RACF 以外のお客様にもご利用いただけます。
- RACF 以外のセキュリティ製品を使用している場合は、セットアップが異なることがあります。詳細については、使用しているセキュリティ製品の資料を参照してください。
- Developer for System z の SMP/E インストールは、内部 RSE プログラムのプログラム制御ビットを設定します。
- プログラム制御ビットの現在の状況を表示するには、z/OS UNIX コマンド **ls -Eog** を使用します (2 番目のストリング内に英字の **p** が表示される場合、そのファイルはプログラムで制御されます)。

```
$ ls -Eog /usr/lib/libIRRacsf.so
-rwxr-xr-x  aps-  2    69632 Oct  5 2007 /usr/lib/libIRRacsf.so
```

セキュリティ設定の検査

セキュリティに関連するカスタマイズの結果を表示するには、以下のサンプル・コマンドを使用します。

- セキュリティの設定とクラス
 - SETROPTS LIST
- ユーザーの OMVS セグメント
 - LISTUSER #userid NORACF OMVS
 - LISTGRP #group-name NORACF OMVS
- データ・セット・プロファイル
 - LISTGRP FEK
 - LISTDSD PREFIX(FEK) ALL
- 開始タスク
 - LISTGRP STCGROUP OMVS
 - LISTUSER STCJMON OMVS
 - LISTUSER STCRSE OMVS
 - LISTUSER STCLOCK OMVS
 - RLIST STARTED JMON.* ALL STDATA
 - RLIST STARTED RSED.* ALL STDATA
 - RLIST STARTED LOCKD.* ALL STDATA
- JES コマンド・セキュリティ
 - RLIST CONSOLE JMON ALL
 - RLIST OPERCMDS MVS.MCSOPER.JMON ALL
 - RLIST OPERCMDS JES%.* ALL

- セキュアな z/OS UNIX サーバーとしての RSE
 - RLIST FACILITY BPX.SERVER ALL
- RSE の MVS プログラム制御ライブラリー
 - RLIST PROGRAM ** ALL
- RSE のアプリケーション保護
 - RLIST APPL FEKAPPL ALL
- RSE の PassTicket サポート
 - RLIST PTKTDATA FEKAPPL ALL SSIGNON
 - RLIST PTKTDATA IRRPTAUTH.FEKAPPL.* ALL
- RSE の z/OS UNIX プログラム制御ファイル
 - ls -E /usr/lib/libIRRracf.so

第 11 章 Developer for System z について

Developer for System z ホストは、クライアントがホスト・サービスとデータにアクセスできるようにするために相互に作用する、複数のコンポーネントで構成されています。これらのコンポーネントの設計を理解しておく、構成に関して適切な判断を行うことができます。

この章では、以下のトピックについて説明します。

- 『コンポーネントの概要』
- 207 ページの『Java アプリケーションとしての RSE』
- 208 ページの『タスク所有者』
- 210 ページの『接続のフロー』
- 212 ページの『ロック・デーモン』
- 214 ページの『z/OS UNIX ディレクトリ構造』

コンポーネントの概要

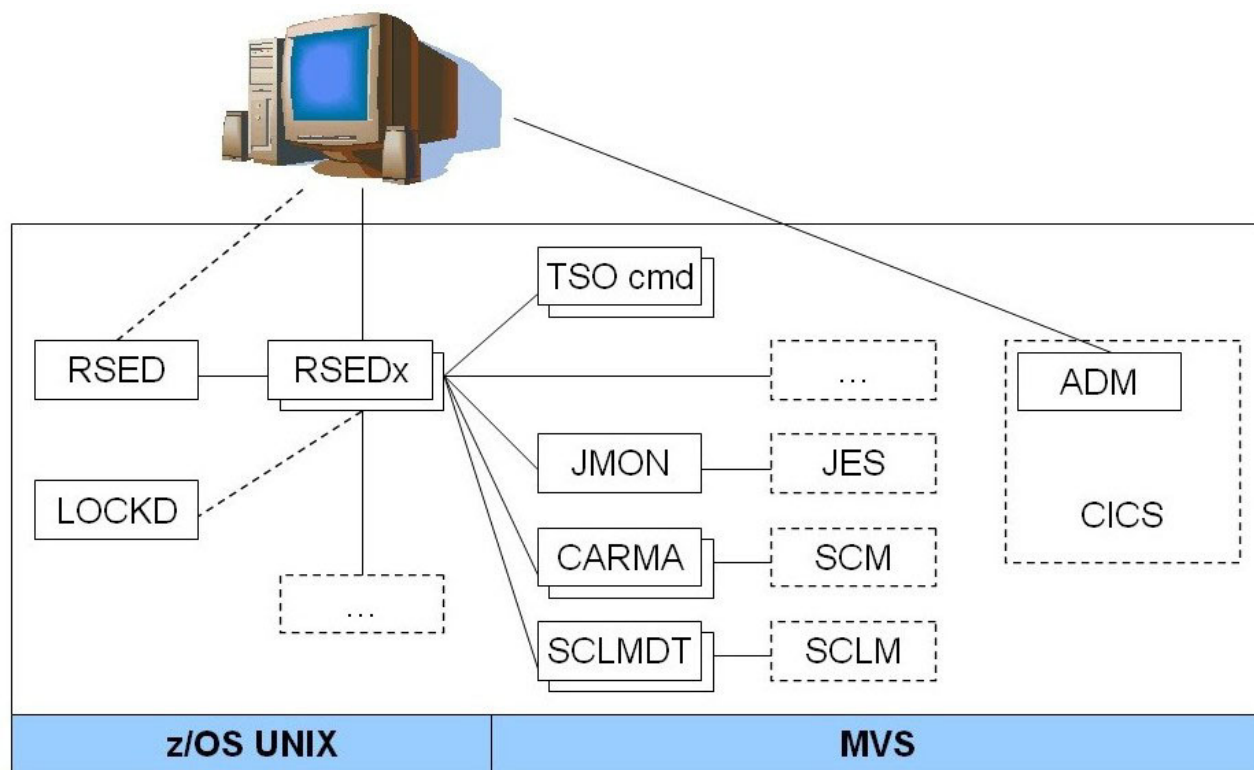


図 41. コンポーネントの概要

図 41 は、ホスト・システムにおける Developer for System z のレイアウトの一般的な概要です。

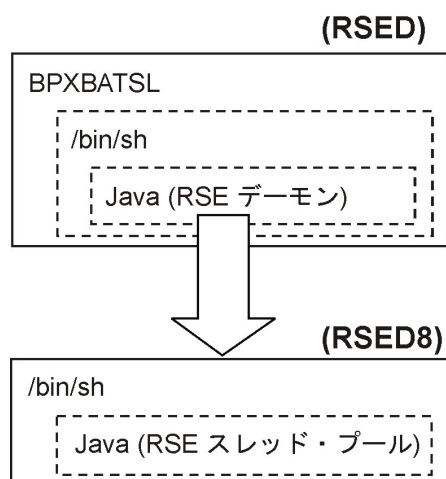
- リモート・システム・エクスプローラー (RSE) は、クライアントをホストに接続したり、特定のサービス用に他のサーバーを始動するなどの、コア・サービスを提供します。RSE は、次の 2 つの論理エンティティから構成されます。
 - RSE デーモン (RSED)。これは接続セットアップを管理します。また、単一サーバー・モードでの実行を担当します。そのために、RSE デーモンは RSE スレッド・プール (RSEDx) と呼ばれる子プロセスを 1 つ以上作成します。
 - RSE サーバー。これは個々のクライアント要求を処理します。RSE サーバーは、RSE スレッド・プール内のスレッドとしてアクティブになります。
- ロック・デーモン (LOCKD) は、データ・セット・ロックのトラッキング・サービスを提供します。
- TSO コマンド・サービス (TSO cmd) は、TSO および ISPF コマンドに、バッチに似たインターフェースを提供します。
- JES ジョブ・モニター (JMON) は、JES に関連したすべてのサービスを提供します。
- 共通アクセス・リポジトリ・マネージャー (CARMA) は、CA Endeavor などの Software Configuration Manager (SCM) と対話するためのインターフェースを提供します。
- SCLM Developer Toolkit (SCLMDT) は、SCLM を拡張し、SCLM と対話するためのインターフェースを提供します。
- Application Deployment Manager (ADM) は、CICS に関連したさまざまなサービスを提供します。
- ほかにも、Developer for System z 自体または相互前提条件のソフトウェアで各種のサービスが提供されています。

前の段落とリストで説明したのは、RSE に割り当てられている中心的な役割です。わずかな例外を除き、クライアント通信はすべて RSE を経由します。これにより、クライアント/ホスト通信に使用されるポートの数が限定されるため、セキュリティに関連したネットワーク・セットアップが容易になります。

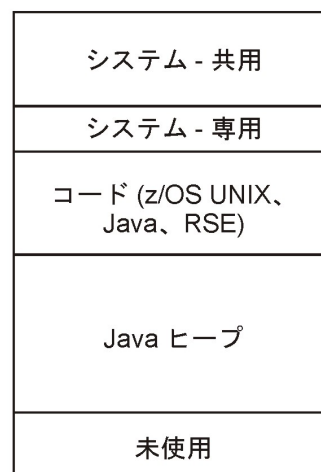
クライアントからの接続とワークロードを管理するために、RSE は、スレッド・プーリング・アドレス・スペースを制御するデーモン・アドレス・スペースから構成されています。デーモンは接続と管理のためのフォーカル・ポイントとして機能し、スレッド・プールはクライアント・ワークロードを処理します。rsed.envvars 構成ファイルに定義されている値と実際のクライアント接続数に基づいて、デーモンは複数のスレッド・プール・アドレス・スペースを開始することができます。

Java アプリケーションとしての RSE

z/OS UNIX プロセス



Java ストレージの使用



JOBNAME	状況	PID	PPID	コマンド
RSED	FILE SYS KERNEL WAIT	50331904	1	BPXBATSL
RSED	WAITING FOR CHILD	67109114	50331904	/bin/sh...
RSED	FILE SYS KERNEL WAIT	50331949	67109114	java...
RSED8	WAITING FOR CHILD	307	50331949	/bin/sh...
RSED8	FILE SYS KERNAL WAIT	308	307	java...

図 42. Java アプリケーションとしての RSE

図 42 は、RSE によるリソースの使用（プロセスとストレージ）を基本的な図で示しています。

RSE は Java アプリケーションであるため、z/OS UNIX 環境でアクティブになります。このため、異なるホスト・プラットフォームへの移植が容易であり、同じく Java アプリケーションである (Eclipse フレームワーク・ベース) Developer for System z クライアントと簡単に通信することができます。したがって、z/OS UNIX および Java の仕組みに関する基本的な知識があると、Developer for System z を理解するうえで非常に役立ちます。

z/OS UNIX では、プログラムが PID (プロセス ID) で識別されるプロセス内で実行されます。各プログラムはそのプログラム専用のプロセス内でアクティブになるため、別のプログラムを呼び出すと新しいプロセスが作成されます。プロセスを開始したプロセスは、PPID (親 PID) で参照され、新しいプロセスは子プロセスと呼ばれます。子プロセスは同じアドレス・スペース内で実行することも、新しいアドレス・スペースで spawn (作成) することもできます。同じアドレス・スペースで新しいプロセスを実行することは、TSO でコマンドを実行することに相当し、新しいアドレス・スペースでプロセスを spawn することは、バッチ・ジョブを実行依頼することと似ています。

プロセスは、単一スレッドの場合とマルチスレッドの場合があるので注意してください。マルチスレッド・アプリケーション (RSEなど) では、個別のアドレス・スペースのとき (より少ないオーバーヘッドによる) と同様に、個々のスレッドがシステム・リソースを得ようと競合します。

上記のプロセスの説明を 207 ページの図 42 の RSE サンプルに対応付けると、次のような流れになります。

1. RSED タスクが開始されると、このタスクが BPXBATSL を実行し、それによって z/OS UNIX が起動され、シェル環境が作成されます - PID 50331904。
2. このプロセスで rsed.sh シェル・スクリプトが実行され、それが別のプロセス (/bin/sh) で稼働します - PID 67109114。
3. このシェル・スクリプトは、rsed.envvars で定義されている環境変数を設定し、RSE デーモンを始動するために必要なパラメーターで Java を実行します - PID 50331949。
4. RSE デーモンが子プロセス (RSED8) で新しいシェルの spawn します - PID 307。
5. このシェルでは、rsed.envvars で定義されている環境変数が設定され、RSE スレッド・プールを始動するために必要なパラメーターで Java が実行されます - PID 308。

RSE などの Java アプリケーションは、ストレージを直接割り振るのではなく、Java メモリー管理サービスを使用します。これらのサービスは、ストレージの割り振り、ストレージの解放、およびガーベッジ・コレクションと同様に、Java ヒープの限度内で機能します。ヒープの最小サイズと最大サイズは、Java 始動時に (暗黙的または明示的に) 定義されます。

つまり、使用可能なアドレス・スペース・サイズを最大限に活用することは、z/OS 用に不定量 (アクティブなスレッド数に依存します) のシステム制御ブロックを保管できる余裕を確保しながら、大きなヒープ・サイズを定義するというバランスを考慮した両立案になります。

タスク所有者

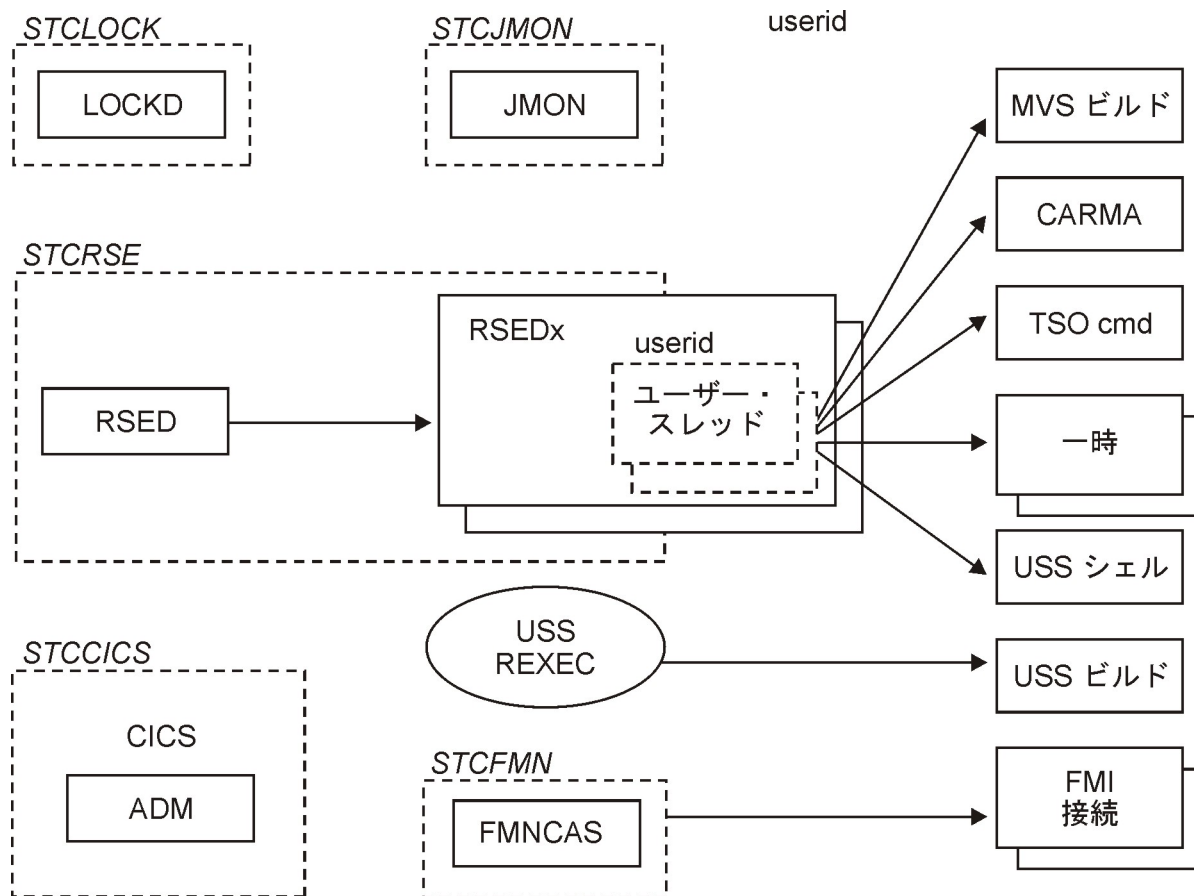


図 43. タスク所有者

図 43 は、Developer for System z のさまざまなタスクで使用するセキュリティー資格情報の所有者の基本的概要を示しています。

タスクの所有権は、2 つの部分に分けることができます。開始タスクは、ご使用のセキュリティー・ソフトウェアで開始タスクに割り当てられているユーザー ID が所有します。それ以外のすべてのタスク (RSE スレッド・プール (RSEDx) は例外) は、クライアント・ユーザー ID が所有します。

図 43 は、Developer for system z の開始タスク (LOCKD、JMON、および RSED)、およびサンプルの開始タスクと Developer for System z が通信するシステム・サービスを示しています。Application Deployment Manager (ADM) が CICS 領域内でアクティブになっています。FMNCAS は、File Manager 開始タスクです。USS REXEC タグは、z/OS UNIX REXEC (または SSH) サービスを表します。

RSE デーモン (RSED) は、クライアント要求を処理するために RSE スレッド・プール・アドレス・スペース (RSEDx) を 1 つ以上作成します。各 RSE スレッド・プールは、複数のクライアントをサポートし、RSE デーモンと同じユーザーによって所有されます。各クライアントには、スレッド・プール内に専用のスレッドがあり、これらのスレッドはクライアント・ユーザー ID が所有します。

クライアントが実行するアクションによっては、1 つ以上の追加のアドレス・スペース (いずれもクライアント・ユーザー ID が所有) を開始して要求されたアクションを実行できます。これらのアドレス・スペースにできるのは、MVS バッチ・ジョ

ブ、APPC トランザクション、または z/OS UNIX 子プロセスです。z/OS UNIX 子プロセスは、z/OS UNIX イニシエーター (BPXAS) 内でアクティブとなり、JES で開始タスクとして表示されることに注意してください。

これらのアドレス・スペースの作成は、ほとんどの場合、スレッド・プール内のユーザー・スレッドによって、直接的に、あるいは ISPF などのシステム・サービスを使用してトリガーされます。ただし、アドレス・スペースはサード・パーティーが作成する可能性もあります。例えば、File Manager は、Developer for System z に代わって処理する必要があるデータ・セット (またはメンバー) ごとに、新しいアドレス・スペースを開始します。z/OS UNIX でビルドを開始する際には、z/OS UNIX REXEC または SSH が関与します。

ユーザー固有のアドレス・スペースは、タスクが完了するか、または非アクティブ・タイマーの期限が切れると終了します。開始タスクはアクティブなままとなります。209 ページの図 43 に示されているアドレス・スペースは、表示の対象となるほど長くシステムに残ります。ただし、z/OS UNIX の設計仕様のために、存続期間の短い一時的なアドレス・スペースもいくつか存在することに注意してください。

接続のフロー

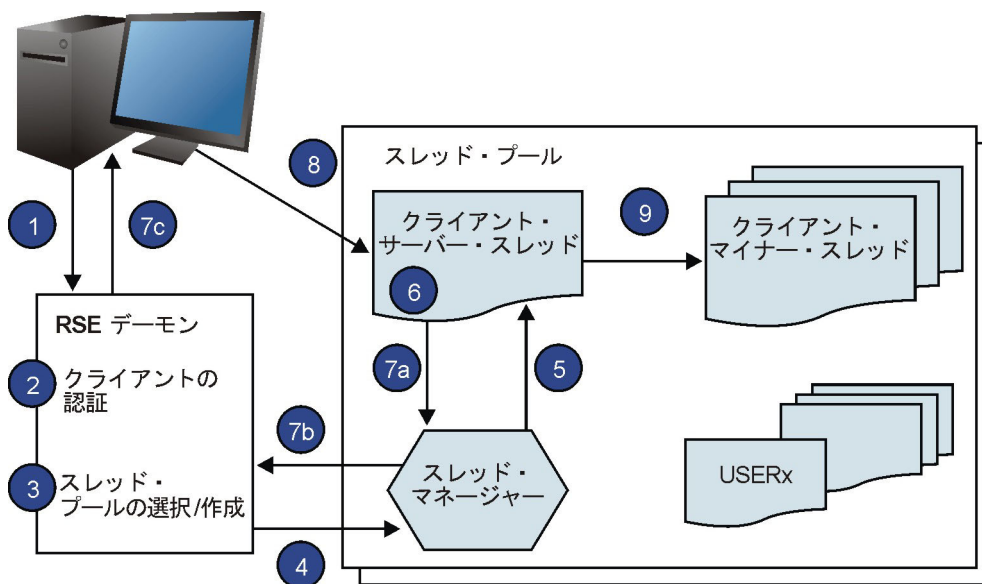


図 44. 接続のフロー

図 44 は、クライアントが Developer for System z を使用してホストに接続する仕組みの概要図です。ここでは、PassTicket の使われ方についても簡単に説明します。

1. クライアントはデーモン (ポート 4035) にログオンします。
2. RSE デーモンは、クライアントから提示された資格情報を使用してクライアントを認証します。
3. RSE デーモンは、既存のスレッド・プールを選択します。すべてのプールが満杯である場合は、新しいスレッド・プールを開始します。

4. RSE デーモンは、スレッド・プールにクライアント・ユーザー ID を渡します。
5. スレッド・プールは、クライアント・ユーザー ID と認証用の PassTicket を使用して、クライアント固有の RSE サーバー・スレッドを作成します。
6. クライアント・サーバー・スレッドは、今後のクライアント通信に使用するポートにバインドします。
7. クライアント・サーバー・スレッドは、クライアントの接続先となるポート番号を返します。
8. クライアントは、RSE デーモンとの接続を解除し、提供されたポート番号に接続します。
9. クライアント・サーバー・スレッドは、他のユーザー固有のスレッド (マイナー) を開始します。その際、常にクライアント・ユーザー ID と認証用の PassTicket を使用します。これらのスレッドから、クライアントが要求したユーザー固有のサービスが提供されます。

上記の説明は、RSE のスレッド指向の設計を示しています。ユーザー単位でアドレス・スペースを開始するのではなく、複数のユーザーが単一のスレッド・プール・アドレス・スペースでサービスされます。スレッド・プール内では、各マイナー (ユーザー固有のサービス) が、ユーザーのセキュリティー・コンテキストが割り当てられたそのマイナー専用のスレッドでアクティブとなるため、セキュアなセットアップが確保されます。この設計は、リソース使用量を制限しながら多数のユーザーに対応しますが、各クライアントが複数のスレッド (実行されるタスクによっては 16 以上) を使用することを実際には意味しています。

ネットワークの観点では、Developer for System z はパッシブ・モードの FTP と同じように動作します。クライアントはフォーカル・ポイント (RSE デーモン) に接続し、その接続をドロップして、フォーカル・ポイントから提供されたポート番号に再接続します。この 2 番目の接続に使用されるポートの選択を制御するロジックを以下に示します。

1. クライアントがサブシステム・プロパティー・タブでゼロ以外のポート番号を指定した場合、RSE サーバーはそのポートをバインドに使用します。このポートが使用不可であると、接続は失敗します。
2. `rsed.envvars` に `_RSE_PORTRANGE` が指定されている場合、RSE サーバーはこの範囲から選択したポートにバインドします。使用可能なポートがない場合、接続は失敗します。RSE サーバーは、クライアントが接続している間そのポートを独占するわけではありません。他の RSE サーバーがそのポートにバインドできないのは、(サーバーの) バインドから (クライアントの) 接続までの間だけです。
3. 制限が設定されていなければ、RSE サーバーはポート 0 にバインドします。その結果、TCP/IP によってポート番号が選択されます。

認証を必要とするすべての z/OS サービスに PassTicket が使用されるので、Developer for System z は、パスワードを保管したりユーザーに毎回パスワードを要求したりすることなく、これらのサービスを任意に呼び出すことができます。また、すべての z/OS サービスに PassTicket を使用することで、ログオン時にワンタイム・パスワードや X.509 証明書などの代替認証方式を使用することもできます。

ロック・デーモン

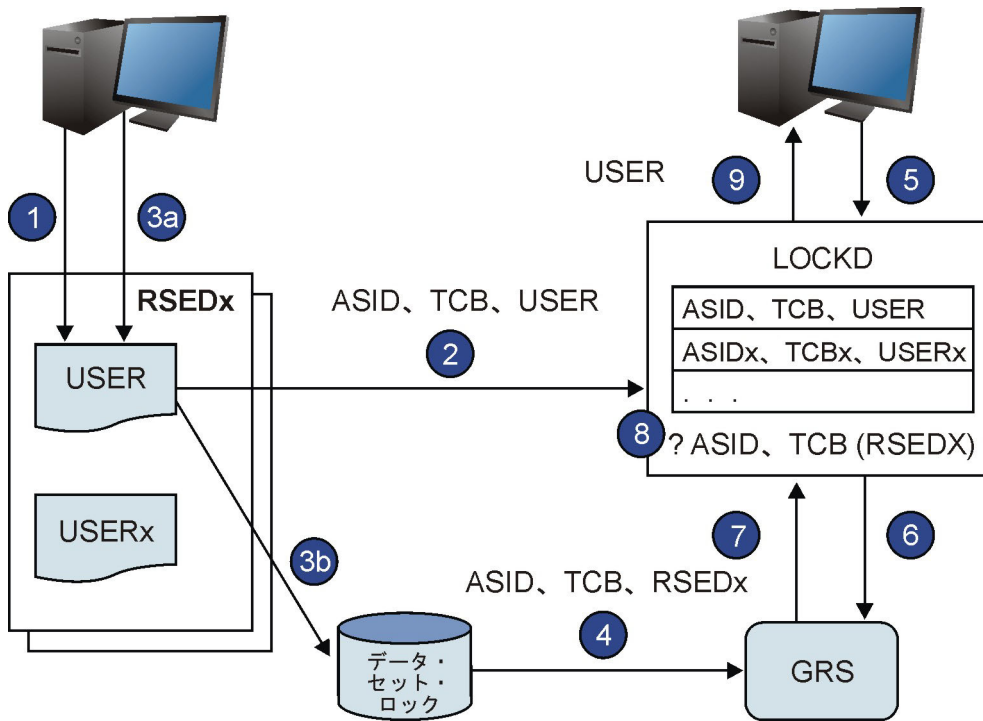


図 45. ロック・デーモンのフロー

図 45 は、データ・セット・ロックを所有する Developer for System z クライアントをロック・デーモンが判別する仕組みの概要図です。

1. クライアントがログオンすると、スレッド・プール (RSEdx) 内にユーザー固有の RSE サーバー・スレッド (USER) が作成されます。
2. RSE サーバーは、新たに接続したユーザーをロック・デーモンに登録します。登録情報には、アドレス・スペース ID (スレッド・プールの ASID)、タスク制御ブロック (TCB) ID (ユーザー固有)、およびユーザー ID が含まれます。
3. クライアントが編集モードでデータ・セットをオープンすると、RSE サーバーはそのデータ・セットに排他ロックを設定するように指示を受けます。
4. システムは、要求側の ASID、TCB、およびタスク名 (RSEdx) をロック・プロセスの一部として登録します。この情報は、グローバル・リソースの逐次化 (GRS) キューに保管されます。
5. オペレーターが (またはクライアントに代わって RSE サーバーが) ロック・デーモンにデータ・セットのロック状況を照会します。
6. ロック・デーモンは GRS キューをスキャンし、データ・セットがロックされているかどうかを確認します。
7. デーモンは、ロック所有者の ASID、TCB、およびタスク名を取り出します。
8. 取り出された ASID と TCB は、登録されているクライアントの ASID および TCB の組み合わせと照合されます。
9. 一致が確認された場合は、関連するクライアント・ユーザー ID が要求側に返されます。そうでない場合は、GRS から取り出されたタスク名が返されます。

複数のユーザーが単一のスレッド・プール・アドレス・スペースに割り当てられる、Developer for System z のシングル・サーバー・セットアップでは、z/OS がデータ・セットまたはメンバーに設定されているロックの所有者を追跡できません。システム・コマンドはアドレス・スペース・レベル、つまりスレッド・プールで停止します。

この問題に対処するために、Developer for System z ではロック・デーモンを提供しています。ロック・デーモンは、RSE ユーザーがデータ・セット/メンバーに設定したすべてのロック、および ISPF などの他の製品が設定したロックを追跡することができます。

RSE サーバーは、新たに接続したユーザーをロック・デーモンに登録します。登録情報には、アドレス・スペース ID (スレッド・プールの ASID)、タスク制御ブロック (TCB) ID (ユーザー固有)、およびユーザー ID が含まれます。

登録は接続時にだけ行われるため、ロック・デーモンが始動 (または再始動) される前にアクティブとなった RSE ユーザーはいずれも登録されないので注意してください。

ロック・デーモンは、データ・セットの照会を受け取ると (modify query オペレーター・コマンドによって、またはクライアントから RSE サーバー経由で)、システムのグローバル・リソースの逐次化 (GRS) キューをスキャンします。ASID と TCB が、登録されているユーザーのものと一致した場合は、そのユーザー ID がロック所有者として返されます。一致しなかった場合は、ASID に関連したジョブ名/ユーザー ID がロック所有者として返されます。

登録が失敗した場合は、コンソール・メッセージ (FEK513W) が登録情報とともに表示されます。これにより、オペレーターが、この値を **DISPLAY GRS,RES=(*,dataset*)** オペレーター・コマンドの出力と照合して、ロック所有者を探することができます。

注: log_level が 2 に設定されている場合は、サーバーの DD STDOUT にも正常に実行された登録が出力されます。これは、ロック・デーモンの再始動後に除去された正常な登録を手動でマッピングする場合に役立ちます。

ロックの解放

通常的环境では、データ・セットまたはメンバーは、クライアントが編集モードでオープンしたときにロックされ、クライアントが編集セッションを終了したときに解放されます。

一定のエラー条件によって、このメカニズムが設計どおりに機能しなくなることがあります。この場合は、ロックを保持しているユーザーを RSE の **modify cancel** オペレーター・コマンドで取り消すことができます。これについては 133 ページの『第 8 章 オペレーター・コマンド』で説明しています。このユーザーに属しているアクティブなデータ・セット・ロックが、プロセスの中で解放されます。

z/OS UNIX ディレクトリー構造

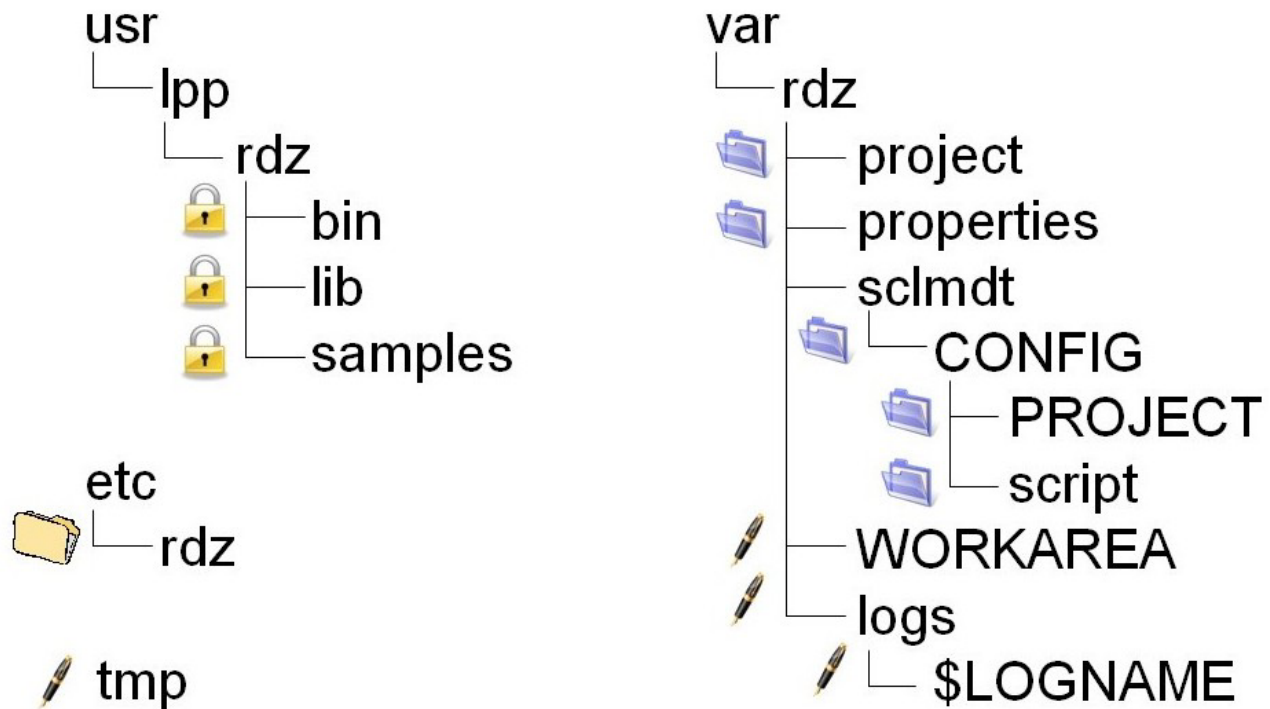


図 46. z/OS UNIX ディレクトリー構造

図 46 は、Developer for System z が使用する z/OS UNIX ディレクトリーの概要を示しています。また、以下のリストでは、Developer for System z が関与する各ディレクトリー、ロケーションの変更方法、および内部のデータを保守する当事者について説明します。

- `/usr/lpp/rdz/` は、Developer for System z 製品コードのルート・パスです。実際のロケーションは、RSED および LOCKD 開始タスク (変数 HOME) で指定されます。この中のファイルは、SMP/E が保守します。
- `/etc/rdz/` には、RSE とマイナーに関連する構成ファイルが保持されます。実際のロケーションは、RSED および LOCKD 開始タスク (変数 CNFG) で指定されます。この中のファイルは、システム・プログラマーが保守します。
- `/var/rdz/sclmdt/CONFIG/` には、汎用の SCLMDT 構成ファイルが保持されます。実際のロケーションは、rsed.envvars (変数 SCLMDT_CONF_HOME) で指定されます。この中のファイルは、SCLM 管理者が保守します。
- `/var/rdz/sclmdt/CONFIG/PROJECT/` には、SCLMDT プロジェクト構成ファイルが保持されます。実際のロケーションは、rsed.envvars (変数 SCLMDT_CONF_HOME) で指定されます。この中のファイルは、SCLM 管理者が保守します。
- `/var/rdz/sclmdt/CONFIG/script/` には、他の製品が使用する SCLMDT 関連のスク립トが保持されます。実際のロケーションはどこにも指定されません。この中のファイルは、SCLM 管理者が保守します。

- `/var/rdz/projects/` には、ホスト・ベースのプロジェクト定義ファイルが保持されます。実際のロケーションは、`projectcfg.properties` (変数 `PROJECT-HOME`) で指定されます。この中のファイルは、プロジェクト・マネージャーまたは主任開発者が保守します。
- `/var/rdz/properties/` には、ホスト・ベースのプロパティ・グループが保持されます。実際のロケーションは、`propertiescfg.properties` (変数 `PROPERTY-GROUP` および `DEFAULT-VALUES`) で指定されます。この中のファイルは、プロジェクト・マネージャーまたは主任開発者が保守します。
- `/var/rdz/logs/` には、RSE デーモンおよび RSE スレッド・プール・サーバーのログが保持されます。実際のロケーションは、`rsed.envvars` (変数 `daemon.log`) で指定されます。この中のファイルは、RSE が保守します。
- `/var/rdz/logs/$LOGNAME/` には、RSE サーバーおよびマイナーのユーザー固有ログが保持されます。実際のロケーションは、`rsed.envvars` (変数 `user.log` および `DSTORE_LOG_DIRECTORY`) で指定されます。この中のファイルは、RSE およびマイナーが保守します。

注: 各クライアントが独自の `$LOGNAME` ディレクトリを作成してユーザー固有のログ・ファイルを保管できるようにするには、`/var/rdz/logs/` に許可ビット・マスク `777` が必要です。

- `/var/rdz/WORKAREA/` は、ISPF の TSO/ISPF クライアント・ゲートウェイおよび SCLMDT が、z/OS UNIX と MVS ベースのアドレス・スペース間でデータを転送するために使用します。実際のロケーションは、`rsed.envvars` (変数 `_CMDSERV_WORK_HOME`) で指定されます。この中のファイルは、ISPF および SCLMDT が保守します。

注: 各クライアントが一時ファイルを作成できるようにするには、`/var/rdz/WORKAREA/` に許可ビット・マスク `777` が必要です。

- `/tmp/` は、ISPF の TSO/ISPF クライアント・ゲートウェイと各種マイナーが一時データを保管するために使用します。ロケーションはカスタマイズできません。この中のファイルは、ISPF およびマイナーが保守します。これは Java ダンプ・ファイルのデフォルト・ロケーションでもあります。このロケーションは、`rsed.envvars` の `_CEE_DUMPTARG` 変数でカスタマイズできます。

注: 各クライアントが一時ファイルを作成できるようにするには、`/tmp/` に許可ビット・マスク `777` が必要です。

非システム管理者の更新特権

`/var/rdz/projects/` などの一部のディレクトリ内のデータは、z/OS UNIX で多数の更新特権を持たないプロジェクト・マネージャーなどの非システム管理者によって保守されます。1 つのユーザー ID だけがファイルを保守している場合は、そのユーザー ID をディレクトリとその中の全データの所有者にすると問題はなくなります。

```
chown -R IBMUSER /var/rdz/projects/
```

複数のユーザー ID がディレクトリへの更新アクセス権を必要とする場合は、グループ許可ビットで対応することができます。

1. 有効な OMVS セグメントを持つグループをご使用のセキュリティー・ソフトウェアで作成し、更新アクセス権を必要とするすべてのユーザー ID を接続します。可能な場合は、これをそれらのユーザー ID のデフォルト・グループにします。

```
ADDGROUP RDZPROJ OMVS(GID(1200))  
CONNECT IBMUSER GROUP(RDZPROJ)  
ALTUSER IBMUSER DFLTGRP(RDZPROJ)
```

2. z/OS UNIX の **chgrp** コマンドを使用して、このグループをディレクトリーとその中の全ファイルに割り当てます。新しいファイルが追加されるときに、目的のグループがそのファイルを追加したユーザー ID のデフォルト・グループでない場合は、ファイルを追加するたびにこのコマンドを繰り返す必要があります。

```
chgrp -R IBMUSER /var/rdz/projects/
```

3. z/OS UNIX の **chmod** コマンドを使用して、グループ全体にディレクトリーとその中の全ファイルに対する更新アクセス権を付与します。

```
chmod -R 775 /var/rdz/projects/
```

第 12 章 WLM に関する考慮事項

従来の z/OS アプリケーションとは異なり、Developer for System z は、ワークロード・マネージャー (WLM) で容易に識別できる一体構造のアプリケーションではありません。Developer for System z は、クライアントがホスト・サービスとデータにアクセスできるようにするために相互に作用する、複数のコンポーネントで構成されています。205 ページの『第 11 章 Developer for System z について』で説明しているように、これらのサービスの一部は異なるアドレス・スペースでアクティブとなるため、WLM 分類も異なることになります。

この章では、以下のトピックについて説明します。

- 『ワークロード分類』
- 219 ページの『目標の設定』

ワークロード分類

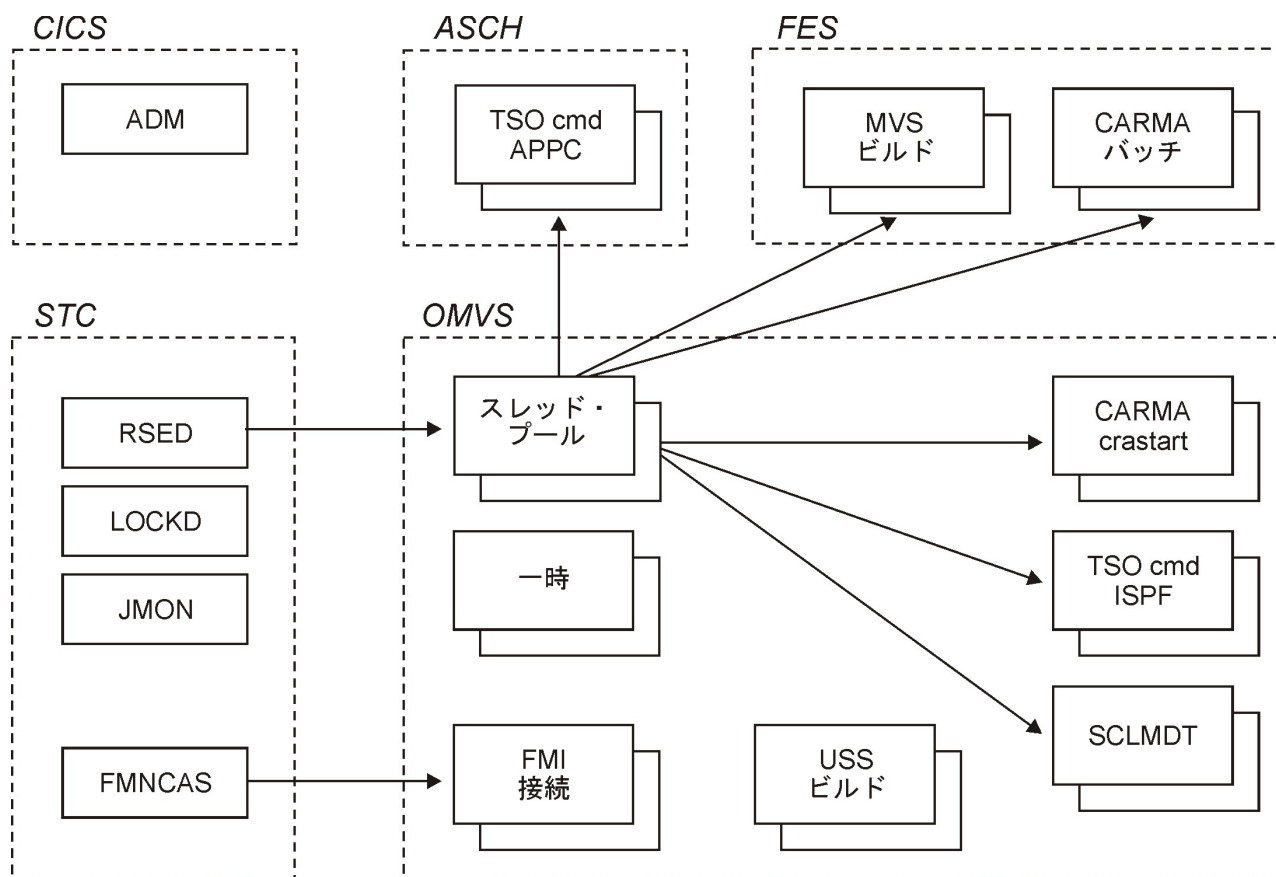


図 47. WLM 分類

図 47 は、Developer for System z ワークロードが WLM に提示されるときに経由するサブシステムの基本的概要を示しています。

Application Deployment Manager (ADM) は CICS 領域でアクティブになるため、WLM での CICS 分類規則に従います。

RSE デーモン (RSED)、ロック・デーモン (LOCKD)、および JES ジョブ・モニター (JMON) は、Developer for System z の開始タスク (または長期実行バッチ・ジョブ) であり、それぞれが専用のアドレス・スペースを使用します。

207 ページの『Java アプリケーションとしての RSE』で説明しているように、RSE デーモンは RSE スレッド・プール・サーバー (不定数のクライアントをサポート) ごとに子プロセスを spawn します。各スレッド・プールは個別のアドレス・スペースでアクティブになります (z/OS UNIX イニシエーター BPXAS を使用)。これらは spawn されたプロセスであるため、開始タスクの分類規則ではなく、WLM OMVS の分類規則を使用して分類されます。

ユーザーが実行するアクションによっては、スレッド・プール内でアクティブなクライアントが他のアドレス・スペースを多数作成する可能性があります。Developer for System z の構成によっては、TSO コマンド・サービス (TSO cmd) や CARMA などの一部のワークロードが、異なるサブシステムで実行される可能性があります。

217 ページの図 47 に示されているアドレス・スペースは、表示の対象となるほど長くシステムに残りますが、z/OS UNIX の設計仕様のために、存続期間の短い一時的なアドレス・スペースもいくつか存在することに注意してください。これらの一時的なアドレス・スペースは、OMVS サブシステム内でアクティブになります。

RSE スレッド・プールは RSE デーモンと同じユーザー ID および同様のジョブ名を使用しますが、スレッド・プールによって開始されるアドレス・スペースはどれも、アクションを要求しているクライアントのユーザー ID によって所有されることに注意してください。このクライアント・ユーザー ID は、スレッド・プールによって開始されるすべての OMVS ベース・アドレス・スペースのジョブ名 (の一部) としても使用されます。

Developer for System z が使用するその他のサービス (File Manager (FMNCAS) や z/OS UNIX REXEC (USS ビルド) など) によって、さらにアドレス・スペースが作成されます。

分類規則

WLM は、分類規則を使用して、システムに入ってきた作業をサービス・クラスにマッピングします。この分類は、作業修飾子に基づいています。最初の (必須) 修飾子は、作業要求を受け取るサブシステム・タイプです。表 29 に、Developer for System z ワークロードを受け取る可能性があるサブシステム・タイプを示します。

表 29. WLM エントリー・ポイント・サブシステム

サブシステム・タイプ	作業の説明
ASCH	作業要求には、IBM 提供の APPC/MVS トランザクション・スケジューラー ASCH によってスケジュールされるすべての APPC トランザクション・プログラムが含まれます。

表 29. WLM エントリー・ポイント・サブシステム (続き)

サブシステム・タイプ	作業の説明
CICS	作業要求には、CICS によって処理されるすべてのトランザクションが含まれます。
JES	作業要求には、JES2 または JES3 が開始するすべてのジョブが含まれます。
OMVS	作業要求には、z/OS UNIX システム・サービスで fork された子のアドレス・スペースで処理される作業が含まれます。
STC	作業要求には、START および MOUNT コマンドによって開始されるすべての作業が含まれます。STC には、システム・コンポーネント・アドレス・スペースも含まれます。

表 30 に、ワークロードを特定のサービス・クラスに割り当てるために使用できる追加の修飾子を示します。リストされている作業修飾子の詳細については、「MVS 計画: ワークロード管理」(SA88-8574) を参照してください。

表 30. WLM 作業修飾子

		ASCH	CICS	JES	OMVS	STC
AI	アカウント情報	x		x	x	x
LU	LU 名 (*)		x			
PF	実行 (*)			x		x
PRI	優先順位			x		
SE	スケジューリング環境名			x		
SSC	サブシステム・コレクション名			x		
SI	サブシステム・インスタンス (*)		x	x		
SPM	サブシステム・パラメーター					x
PX	シスプレックス名	x	x	x	x	x
SY	システム名 (*)	x			x	x
TC	トランザクション/ジョブ・クラス (*)	x		x		
TN	トランザクション/ジョブ名 (*)	x	x	x	x	x
UI	ユーザー ID (*)	x	x	x	x	x

注: (*) のマークが付いた修飾子については、タイプの省略形に G を付加することで、分類グループを指定できます。例えば、トランザクション名グループは TNG となります。

目標の設定

217 ページの『ワークロード分類』で説明しているように、Developer for System z はシステム上でさまざまなタイプのワークロードを作成します。これらの各種タスクは互いに通信します。つまり、タスク間接続でのタイムアウトの問題を回避するためには、実際の経過時間が重要になります。このため、Developer for System z の

タスクは、ハイパフォーマンスのサービス・クラスに配置するか、または優先順位の高い適度なパフォーマンスのサービス・クラスに配置する必要があります。

したがって、現行の WLM の目標を改訂または更新することをお勧めします。これは特に、従来の MVS 作業現場で時間依存型の OMVS ワークロードを初めて扱う場合に当てはまります。

注:

- このセクションに記載する目標の情報は、あえて説明レベルにとどめていますが、これは、実際のパフォーマンス目標がサイトによって大きく異なるためです。
- システムでの特定のタスクの影響を理解しやすくするために、最少のリソース使用量、中程度のリソース使用量、および相当なリソース使用量といった言葉を使用しています。これらはいずれも、システム全体ではなく Developer for System z の総リソース使用量を基準としています。

表 31 は、Developer for System z が使用するアドレス・スペースを示しています。z/OS UNIX では、「タスク名」列の「x」がランダムな 1 桁の数値で置き換えられます。

表 31. WLM ワークロード

説明	タスク名	ワークロード
JES ジョブ・モニター	JMON	STC
ロック・デーモン	LOCKD	STC
RSE デーモン	RSED	STC
RSE スレッド・プール	RSEDx	OMVS
ISPF クライアント・ゲートウェイ (TSO コマンド・サービスおよび SCLMDT)	<userid>x	OMVS
TSO コマンド・サービス (APPC)	FEKFRSRV	ASCH
CARMA (パッチ)	CRA<port>	JES
CARMA (crastart)	<userid>x	OMVS
CARMA (ISPF クライアント・ゲートウェイ)	<userid> および <userid>x	OMVS
MVS ビルド (パッチ・ジョブ)	*	JES
z/OS UNIX ビルド (シェル・コマンド)	<userid>x	OMVS
z/OS UNIX シェル	<userid>	OMVS
File Manager タスク	<userid>x	OMVS
Application Deployment Manager	CICSTS	CICS

目標の選択に関する考慮事項

以下に示す WLM の一般的な考慮事項は、Developer for System z に対して適切に目標を定義するために役立ちます。

- 望ましい結果ではなく、実際に達成できることに基いて目標を設定してください。目標を必要以上に高く設定すると、WLM は重要性の低い作業から重要性の高い作業にリソースを移しますが、この重要性の高い作業が実際にはリソースを必要としない場合があります。

- **SYSTEM** および **SYSSTC** サービス・クラスに割り当てられる作業量を制限してください。これは、これらのクラスのディスパッチング優先順位が **WLM** 管理クラスよりも高いためです。この 2 つのクラスは、重要性が高く、かつ **CPU** の使用量が少ない作業に使用してください。
- 分類規則の対象から外れた作業は、任意の目標を持つ **SYSOTHER** クラスに割り当てられることになります。任意の目標では、**WLM** に対してシステムに予備のリソースがあるときに最善策を取ることだけが指示されます。

応答時間目標を使用する場合:

- **WLM** で応答時間目標を適切に管理するには、タスクの到着率が一定であることが必要です (少なくとも 20 分間に 10 タスク)。
- 平均応答時間の目標は、十分に制御されたワークロードにのみ使用してください。1 つの長いトランザクションが平均応答時間に大きく影響し、**WLM** が過剰反応する可能性があるからです。

速度目標を使用する場合:

- さまざまな理由から、通常は速度目標の達成率が 90% を超えることは不可能です。例えば、**SYSTEM** および **SYSSTC** アドレス・スペースのディスパッチング優先順位は、すべて速度タイプの目標を上回っています。
- **WLM** は、その速度目標を、最小限の数の (使用および遅延) サンプルに基づいて決定します。このため、サービス・クラスで実行されている作業が少ないほど、必要な数のサンプルの収集とディスパッチング・ポリシーの調整に時間がかかることになります。
- ハードウェアを変更する場合は、速度目標を再評価してください。特に、より少ない台数のより高速なプロセッサへと移行する場合は、速度目標の変更が必要となります。

STC

Developer for System z の開始タスクである **RSE** デーモン、ロック・デーモン、および **JES** ジョブ・モニターはいずれも、リアルタイムのクライアント要求を処理します。

表 32. **WLM** ワークロード - **STC**

説明	タスク名	ワークロード
JES ジョブ・モニター	JMON	STC
ロック・デーモン	LOCKD	STC
RSE デーモン	RSED	STC

- **JES** ジョブ・モニター

JES ジョブ・モニターは、ジョブの実行依頼、スプール・ファイルの表示、**JES** オペレーター・コマンドの実行など、すべての **JES** 関連サービスを提供します。ハイパフォーマンスの 1 期間の速度目標を指定する必要があります。これは、タスクが **WLM** に個々のトランザクションを報告しないためです。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、最少から中程度と予想されます。

- ロック・デーモン

ロック・デーモンは、クライアントおよびオペレーターから要求があると GRS エンキュー・テーブルを照会し、その結果を既知の Developer for System z ユーザーと突き合わせます。ハイパフォーマンスの 1 期間の速度目標を指定する必要があります。これは、タスクが WLM に個々のトランザクションを報告しないためです。リソース使用量は最少と予想されます。

- RSE デーモン

RSE デーモンは、クライアントのログオンと認証を処理し、複数の RSE スレッド・プールを管理します。ハイパフォーマンスの 1 期間の速度目標を指定する必要があります。これは、タスクが WLM に個々のトランザクションを報告しないためです。リソース使用量は、作業日の開始点をピークとして、中程度と予想されます。

OMVS

OMVS ワークロードは 2 つのグループ、つまり RSE スレッド・プールとそれ以外に分けることができます。これは、RSE スレッド・プールを除くすべてのワークロードが、クライアント・ユーザー ID をアドレス・スペース名のベースとして使用するためです (z/OS UNIX では、「タスク名」列の「x」がランダムな 1 桁の数値で置き換えられます)。

表 33. WLM ワークロード - OMVS

説明	タスク名	ワークロード
RSE スレッド・プール	RSEDx	OMVS
ISPF クライアント・ゲートウェイ (TSO コマンド・サービスおよび SCLMDT)	<userid>x	OMVS
CARMA (crastart)	<userid>x	OMVS
CARMA (ISPF クライアント・ゲートウェイ)	<userid> および <userid>x	OMVS
z/OS UNIX ビルド (シェル・コマンド)	<userid>x	OMVS
z/OS UNIX シェル	<userid>	OMVS
File Manager タスク	<userid>x	OMVS

- RSE スレッド・プール

RSE スレッド・プールは、Developer for System z の中枢であると言えます。ほぼすべてのデータがスレッド・プールを通過し、スレッド・プール内のマイナー (ユーザー固有のスレッド) によって、他の Developer for System z 関連タスクのほとんどが制御されます。ハイパフォーマンスの 1 期間の速度目標を指定する必要があります。これは、タスクが WLM に個々のトランザクションを報告しないためです。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、相当量と予想されます。

残りのワークロードは、アドレス・スペース命名規則が共通であるために、すべて同じサービス・クラスに割り当てられることになります。このサービス・クラスには、複数期間の目標を指定する必要があります。最初の期間にはハイパフォーマンスのパーセンタイル応答時間目標を指定し、最後の期間には適度なパフォーマンス

の速度目標を指定する必要があります。ISPF クライアント・ゲートウェイなどの一部のワークロードは、個々のトランザクションを WLM に報告しますが、それ以外のワークロードはこれを行いません。

- ISPF クライアント・ゲートウェイ

ISPF クライアント・ゲートウェイは、Developer for System z が非対話式の TSO コマンドと ISPF コマンドを実行するために呼び出す ISPF サービスです。これには、クライアントが発行する明示的なコマンドと、Developer for System z が発行する暗黙的なコマンド (PDS メンバー・リストの取得など) が含まれます。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、最少と予想されます。

- CARMA

CARMA はオプションの Developer for System z サーバーで、CA Endevor[®] SCM などのホスト・ベースの Software Configuration Managers (SCM) と対話するために使用されます。Developer for System z では、CARMA サーバーをさまざまな方式で始動することができ、その一部は OMVS ワークロードになります。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、最少と予想されます。

- z/OS UNIX ビルド

クライアントが z/OS UNIX プロジェクトのビルドを開始すると、z/OS UNIX REXEC (または SSH) によって、ビルドを実行するための多数の z/OS UNIX シェル・コマンドを実行するタスクが開始されます。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、プロジェクトのサイズに応じて中程度から相当量と予想されます。

- z/OS UNIX シェル

このワークロードは、クライアントによって発行される z/OS UNIX シェル・コマンドを処理します。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、最少と予想されます。

- IBM File Manager

spawn された File Manager 子プロセスは、Developer for System z アドレス・スペースではありませんが、Developer for System z クライアントの要求に応じて開始でき、これらのタスクが Developer for System z タスクと同じ命名規則を使用することから、ここにリストしています。これらの File Manager タスクは、重要な MVS データ・セット・アクション (VSAM ファイルのフォーマット編集など) を処理します。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、最少から中程度と予想されます。

JES

JES で管理されるバッチ処理は、Developer for System z によってさまざまに使用されます。最も一般的な用途は、MVS ビルドです。ここでは、ジョブが実行依頼され、終了のタイミングを判別するためにモニターされます。ただし、Developer for System z は、CARMA サーバーをバッチで始動し、TCP/IP を使用してそのサーバーと通信することもできます。

表 34. WLM ワークロード - JES

説明	タスク名	ワークロード
CARMA (バッチ)	CRA<port>	JES
MVS ビルド (バッチ・ジョブ)	*	JES

• CARMA

CARMA はオプションの Developer for System z サーバーで、CA Endeavor[®] SCM などのホスト・ベースの Software Configuration Managers (SCM) と対話するために使用されます。Developer for System z では、CARMA サーバーをさまざまな方法で始動することができ、その一部は JES ワークロードになります。ハイパフォーマンスの 1 期間の速度目標を指定する必要があります。これは、タスクが WLM に個々のトランザクションを報告しないためです。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、最少と予想されます。

• MVS ビルド

クライアントが MVS プロジェクトのビルドを開始すると、Developer for System z によって、ビルドを実行するためのバッチ・ジョブ開始されます。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、プロジェクトのサイズに応じて中程度から相当量と予想されます。ご使用のローカル環境に応じて、適度のパフォーマンスを目標とするさまざまな戦略をお勧めできます。

- パーセンタイル応答時間目標の期間と、後続の速度目標の期間で構成される、複数期間の目標を指定できます。この場合、開発者は、応答時間が均一のジョブを作成するために、ほぼ同じビルド・プロシージャとほぼ同じサイズの入力ファイルを使用する必要があります。WLM で応答時間目標を適切に管理するには、ジョブの到着率が一定であることが必要です (少なくとも 20 分間に 10 ジョブ)。
- 速度目標は、実行時間と到着率にかなりのばらつきがあっても対応できるため、ほとんどのバッチ・ジョブに適しています。

ASCH

現行の Developer for System z バージョンでは、ISPF クライアント・ゲートウェイを使用して、非対話式の TSO コマンドと ISPF コマンドが実行されます。歴史的な理由から、Developer for System z は APPC トランザクションによるこれらのコマンドの実行もサポートしています。

表 35. WLM ワークロード - ASCH

説明	タスク名	ワークロード
TSO コマンド・サービス (APPC)	FEKFRSRV	ASCH

• TSO コマンド・サービス

TSO コマンド・サービスは、非対話式の TSO コマンドと ISPF コマンドを実行するために、Developer for System z によって APPC トランザクション・プログラムとして開始することができます。これには、クライアントが発行する明示的

なコマンドと、Developer for System z が発行する暗黙的なコマンド (PDS メンバー・リストの取得など) が含まれます。このサービス・クラスには、複数期間の目標を指定する必要があります。最初の期間には、ハイパフォーマンスのパークセンタイル応答時間目標を指定してください。最後の期間には適度なパフォーマンスの速度目標を指定してください。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、最少と予想されます。

CICS

Application Deployment Manager は、CICS Transaction Server 領域内でアクティブになるオプションの Developer for System z サーバーです。

表 36. WLM ワークロード - CICS

説明	タスク名	ワークロード
Application Deployment Manager	CICSTS	CICS

- Application Deployment Manager

CICSTS 領域でアクティブになるオプションの Application Deployment Manager サーバーでは、選択した CICSTS 管理タスクを開発者に安全にオフロードすることができます。リソース使用量は、ユーザー・アクションに大きく依存するため変動しますが、最少と予想されます。使用すべきサービス・クラスのタイプは、当該 CICS 領域内でアクティブな他のトランザクションによって異なるため、ここでは詳しく説明しません。

WLM は、CICS に使用できる複数の管理タイプをサポートしています。

- 領域目標に向けた CICS の管理

CICS アドレス・スペースを管理するサービス・クラスに目標が設定されます。ユーザーは、このサービス・クラスの実行速度目標のみを使用できます。WLM は、アドレス・スペースに対する JES または STC の分類規則を使用しますが、トランザクションに対する CICS サブシステムの分類規則は使用しません。

- トランザクション応答時間目標に向けた CICS の管理

単一のトランザクションまたはトランザクションのグループに割り当てられたサービス・クラスにおいて、応答時間目標を設定することができます。WLM は、アドレス・スペースに対する JES または STC の分類規則、およびトランザクションに対する CICS サブシステムの分類規則を使用します。

第 13 章 チューニングに関する考慮事項

205 ページの『第 11 章 Developer for System z について』で説明したように、RSE (リモート・システム・エクスプローラー) は、Developer for System z の中核です。クライアントからの接続とワークロードを管理するために、RSE はスレッド・プール・アドレス・スペースを制御するデーモン・アドレス・スペースで成り立っています。デーモンは接続と管理のためのフォーカル・ポイントとして機能し、スレッド・プールはクライアント・ワークロードを処理します。

このため、RSE は Developer for System z セットアップをチューニングする場合の主要な対象となります。ただし、それぞれが 16 個以上のスレッドを使用する何百人ものユーザー、ある程度の大きさのストレージ、そして場合によっては 1 つ以上のアドレス・スペースを保守するには、Developer for System z と z/OS の両方を適切に構成する必要があります。

この章では、以下のトピックについて説明します。

- 『リソース使用量』
- 239 ページの『ストレージの使用量』
- 244 ページの『z/OS UNIX ファイル・システム・スペースの使用量』
- 247 ページの『主要なリソース定義』
- 250 ページの『さまざまなリソース定義』
- 252 ページの『モニター』
- 256 ページの『サンプル・セットアップ』

リソース使用量

このセクションの情報を使用して、Developer for System z の標準のリソース使用量と最大のリソース使用量を見積もり、それに合わせてシステム構成を計画することができます。

このセクションで示す数値と数式を使用してシステム限度を定義するときには、使用する見積もり値がきわめて正確であることに留意してください。システム限度を設定するときは、一時タスクやその他のタスク、あるいは同じホストに同時に複数回接続する (例えば RSE と TN3270 経由で) ユーザーがリソースを使用できるように、十分なゆとりを残すようにします。

注:

- ここで示す情報の範囲は、Developer for System z 自体が提供し RSE 経由でアクセスされるサービスに限定されます。例えば、TN3270 のリソース使用量 (RSE 経由でアクセスされません) や、MVS または z/OS UNIX プロジェクトのリモート (ホスト・ベースの) ビルドで呼び出されるプログラムのリソース使用量 (Developer for System z から提供されません) については記載していません。

- Developer for System z にサード・パーティーの拡張を追加すると、リソース使用量のカウンターが増えることがあります。
- どのサービスにも短時間で終了する「ハウスキーピング」タスクがあります。これらは、その実行中にリソースを使用し、互いに順次または並列に実行されます。これらのタスクが使用するリソースについては記載していません。
- ISPF クライアント・ゲートウェイなどの必要なソフトウェアによるユーザー固有のリソース使用量については、有用な情報のみを記載しています。
- ここに示す数値は、事前の通知なしに変更される可能性があります。

概説

以下の各表は、Developer for System z で使用されるアドレス・スペース、プロセス、およびスレッドの数の概要です。ここに示す数値については、次のセクションで詳しく説明します。

- 229 ページの『アドレス・スペースの数』
- 232 ページの『プロセスの数』
- 235 ページの『スレッドの数』

表 37 は、Developer for System z の開始タスクで使用される主要なリソースの概要です。これらのリソースは、1 回だけ割り振られ、すべての Developer for System z クライアント間で共有されます。

表 37. 共通のリソース使用量

開始タスク	アドレス・スペース数	プロセス数	スレッド数
JMON	1	1	3
LOCKD	1	3	10
RSED	1	3	11
RSEDx	(a)	2	10

注: (a) 少なくとも 1 つの RSE スレッド・プール・アドレス・スペースがアクティブです。RSE スレッド・プール・アドレス・スペースの実際の数を確認するには、229 ページの『アドレス・スペースの数』を参照してください。

表 38 は、必要なソフトウェアで使用される主要なリソースの概要です。これらのリソースは、関連機能呼び出す Developer for System z クライアントごとに割り振られます。

表 38. ユーザーごとに必要なリソース使用量

必要なソフトウェア	アドレス・スペース数	プロセス数	スレッド数
ISPF クライアント・ゲートウェイ	1	2	4
APPC	1	1	2
File Manager	1	1	2

表 39 は、各 Developer for System z クライアントで指定の機能を実行するとき
に使用される主要なリソースの概要です。数値以外の値 (ISPF など) は、228 ページ
の表 38 の対応する値を指しています。

表 39. ユーザーごとのリソース使用量

ユーザーのアクション	アドレス・スペース数	プロセス数	スレッド数		
	ユーザー ID	ユーザー ID	ユーザー ID	RSEDx	JMON
ログオン	-	-	-	16	1
アイドル・タイムアウト用のタイマー	-	-	-	1	-
PDS(E) の拡張	ISPF	ISPF	ISPF	-	-
データ・セットのオープン	ISPF	ISPF	ISPF	-	-
TSO コマンド	ISPF	ISPF	ISPF	-	-
z/OS UNIX シェル	1	1	1	6	-
MVS ビルド	1	-	-	-	-
z/OS UNIX ビルド	3	3	3	-	-
CARMA (バッチ)	1	1	2	1	-
CARMA (crastart)	1	1	2	4	-
CARMA (ispf)	4	4	7	5	-
SCLMDT	ISPF	ISPF	ISPF	-	-
File Manager Integration	ISPF + FM	ISPF + FM	ISPF + FM	-	-
Fault Analyzer Integration	-	-	-	-	-

注: SCLM Developer Toolkit 以外は、ISPF を APPC に置き換えることもできます。

アドレス・スペースの数

表 40 は、Developer for System z で使用されるアドレス・スペース数を示しています。この表の「数」列の「u」は、その機能を使用する同時にアクティブなユーザーの数で値を乗算する必要があることを示します。「タスク名」列の「x」は、z/OS UNIX によってランダムな 1 桁の数値で置き換えられます。

表 40. アドレス・スペースの数

数	説明	タスク名	共用	終了のタイミング
1	JES ジョブ・モニター	JMON	はい	なし

表 40. アドレス・スペースの数 (続き)

数	説明	タスク名	共用	終了のタイミング
1	ロック・デーモン	LOCKD	はい	なし
1	RSE デーモン	RSED	はい	なし
(a)	RSE スレッド・プール	RSEDx	はい	なし
1u	ISPF クライアント・ゲートウェイ (TSO コマンド・サービスおよび SCLMDT)	<userid>x	いいえ	15 分後またはユーザー・ログオフ後
1u	TSO コマンド・サービス (APPC)	FEKFRSRV	いいえ	60 分後またはユーザー・ログオフ後
1u	CARMA (バッチ)	CRA<port>	いいえ	7 分後またはユーザー・ログオフ後
1u	CARMA (crastart)	<userid>x	いいえ	7 分後またはユーザー・ログオフ後
4u	CARMA (ispf)	(1)<userid> または (3)<userid>x	いいえ	7 分後またはユーザー・ログオフ後
(b)	1 人のユーザーによる ISPF クライアント・ゲートウェイの同時使用	<userid>x	いいえ	タスク完了後
1u	MVS ビルド (バッチ・ジョブ)	*	いいえ	タスク完了後
3u	z/OS UNIX ビルド (シェル・コマンド)	<userid>x	いいえ	タスク完了後
1u	z/OS UNIX シェル	<userid>	いいえ	ユーザー・ログオフ後
(c)	File Manager	<userid>x	いいえ	タスク完了後

注:

- (a) 少なくとも 1 つの RSE スレッド・プール・アドレス・スペースがアクティブです。実際数は以下の要素に依存します。
 - `rsed.envvars` 内の `minimum.threadpool.process` ディレクティブ。デフォルト値は 1 です。
 - 1 つのスレッド・プールでサービスできるユーザーの数。デフォルトの設定では、スレッド・プール当たり 60 ユーザーです。
 - 同時にアクティブなユーザー数の最高水準点。これは、使用されていないスレッド・プールが自動的に停止されないためです。
- (b) Developer for System z では、1 人のユーザーに対して複数のスレッドがアクティブになります。ISPF クライアント・ゲートウェイのアドレス・スペースが、あるスレッドの要求を処理している間に別のスレッドが要求を送信すると、ISPF は新しいクライアント・ゲートウェイを開始して新しい要求を処理します。このアドレス・スペースは、タスク完了後に終了します。
- (c) File Manager リスナーは、操作が必要なオブジェクト (VSAM など) ごとにアドレス・スペースを開始します。このアドレス・スペースは、オブジェクトが不要となったことが Developer for System z から通知される (VSAM を終了するなどして) までアクティブです。

- SCLMDT は、ISPF クライアント・ゲートウェイ・アドレス・スペースを必要とします。SCLMDT は、このアドレス・スペースを TSO コマンド・サービスと共有します。
- MVS データ・セット関連アクションのほとんどは TSO コマンド・サービスを使用しますが、このサービスは個々に ISPF クライアント・ゲートウェイまたは APPC トランザクションでアクティブとなります。

図 48 の数式は、Developer for System z で使用されるアドレス・スペースの最大数を見積もる場合に使用します。

$$3 + A + N*(x + y + z) + (2 + N*0.01)$$

図 48. アドレス・スペースの最大数

各項の説明は次のとおりです。

- 「3」は、永続的にアクティブなサーバー・アドレス・スペースの数です。
- 「A」は、RSE スレッド・プール・アドレス・スペースの数を表します。
- 「N」は、同時ユーザーの最大数を表します。
- 「x」は、選択した構成オプションに応じて、以下のいずれかの値になります。

X	SCLMDT	クライアント・ゲートウェイ経由の TSO	APPC 経由の TSO
1	いいえ	いいえ	はい
1	いいえ	はい	いいえ
1	はい	はい	いいえ

- 「y」は、選択した構成オプションに応じて、以下のいずれかの値になります。

y	
0	CARMA なし
1	CARMA (バッチ)
1	CARMA (crastart)
4	CARMA (ispf)

- 「z」はデフォルトでは 0 ですが、ユーザーのアクションによっては増える可能性があります。
 - MVS ビルドが実行される場合は 1 を加算します。これらのアドレス・スペースは、関連するビルド・タスク (バッチ・ジョブ) が完了した時点で終了します。
 - z/OS UNIX ビルドが実行される場合は 3 を加算します。呼び出されたプログラムの必要性によっては、実際の数値がこれより大きくなる場合があるので注意してください。これらのアドレス・スペースは、関連するビルド・タスクが完了した時点で終了します。
 - IBM File Manager との同時対話ごとに 1 を加算します。これらのアドレス・スペースは、要求されたオブジェクトが不要になった時点で終了します。

- ・「 $2 + N \cdot 0.01$ 」は、一時アドレス・スペース用のバッファを追加します。必要なバッファ・サイズは、ご使用のサイトによって異なる場合があります。

図 49 の数式は、Developer for System z クライアントで使されるアドレス・スペースの最大数を見積もる場合に使用します (ここに記載していない一時アドレス・スペースはカウントしていません)。

$$x + y + z$$

図 49. クライアントごとのアドレス・スペース数

各項の説明は次のとおりです。

- ・「 x 」は、選択した構成オプションによって異なります。アドレス・スペースの最大数を計算する数式 (231 ページの図 48) を参照してください。
- ・「 y 」は、選択した構成オプションによって異なります。アドレス・スペースの最大数を計算する数式 (231 ページの図 48) を参照してください。
- ・「 z 」はデフォルトでは 0 ですが、ユーザーのアクションによっては増える可能性があります。アドレス・スペースの最大数を計算する数式 (231 ページの図 48) を参照してください。

表 41 に示す定義によって、アドレス・スペースの実際の数制限することができません。

表 41. アドレス・スペースの限度

ロケーション	限度	影響を受けるリソース
rsed.envvars	maximum.threadpool.process	RSE スレッド・プールの数を制限します。
IEASYMxx	MAXUSER	アドレス・スペースの数を制限します。
ASCHPMxx	MAX	TSO コマンド・サービス (APPC) での APPC イニシエーターの数を制限します。

プロセスの数

表 42 は、Developer for System z で使されるアドレス・スペース当たりのプロセス数を示しています。この表の「アドレス・スペース数」列の「 u 」は、その機能を使用する同時にアクティブなユーザーの数で値を乗算する必要があることを示します。

表 42. プロセスの数

プロセス数	アドレス・スペース数	説明	ユーザー ID
1	1	JES ジョブ・モニター	STCJMON
3	1	ロック・デーモン	STCLOCK
3	1	RSE デーモン	STCRSE
2	(a)	RSE スレッド・プール	STCRSE
2	(b)	ISPF クライアント・ゲートウェイ (TSO コマンド・サービスおよび SCLMDT)	<userid>

表 42. プロセスの数 (続き)

プロセス数	アドレス・スペース数	説明	ユーザー ID
1	1u	TSO コマンド・サービス (APPC)	<userid>
1	1u	CARMA (バッチ)	<userid>
1	1u	CARMA (crastart)	<userid>
1	1u	CARMA (ispf)	<userid>
1	3u	z/OS UNIX ビルド (シェル・コマンド)	<userid>
1	1u	z/OS UNIX シェル	<userid>
1	(c)	File Manager	<userid>
(5)	(u)	SCLM Developer Toolkit	<userid>

注:

- (a) 少なくとも 1 つの RSE スレッド・プール・アドレス・スペースがアクティブです。RSE スレッド・プール・アドレス・スペースの実際の数を確認するには、229 ページの『アドレス・スペースの数』を参照してください。
- RSE デモンおよびすべての RSE スレッド・プールは、同じユーザー ID を使用します。
- (b) 通常の状態デフォルトの構成オプションを使用しているときは、ユーザーごとに 1 つの ISPF クライアント・ゲートウェイがアクティブになります。実際数は変わる可能性があります (229 ページの『アドレス・スペースの数』を参照してください)。
- (c) File Manager リスナーは、操作が必要なオブジェクト (VSAM など) ごとに 1 つのプロセスを使用します。このプロセスは、オブジェクトが不要となったことが Developer for System z から通知される (VSAM を終了するなどして) までアクティブです。
- SCLMDT は、ISPF クライアント・ゲートウェイ・アドレス・スペースを必要とします。SCLMDT は、このアドレス・スペースを TSO コマンド・サービスと共用します。
- (u) SCLMDT プロセスは ISPF クライアント・ゲートウェイのアドレス・スペースで実行されるため、アドレス・スペース数の値はありません。
- SCLMDT プロセスは一時的であり、タスクが完了すると終了しますが、1 人のユーザーに対して同時に複数のプロセスがアクティブになることがあります。232 ページの表 42 には、同時 SCLMDT プロセスの最大数を示しています。
- MVS データ・セット関連アクションのほとんどは TSO コマンド・サービスを使用しますが、このサービスは個々に ISPF クライアント・ゲートウェイまたは APPC トランザクションでアクティブとなります。
- z/OS UNIX ビルドは合計で 3 つのプロセスを使用し、それぞれが専用のアドレス・スペースで実行されます。
- 特に断りがない限り、リストされているすべてのプロセスは、関連するアドレス・スペースが終了するまでアクティブです。

図 50 の数式は、Developer for System z で使用されるプロセスの最大数を見積もる場合に使用します。

$$7 + 2 * A + N * (x + y + z) + (10 + N * 0.05)$$

図 50. プロセスの最大数

各項の説明は次のとおりです。

- ・「7」は、永続的にアクティブなサーバー・アドレス・スペースで使用されるプロセスの数です。
- ・「A」は、RSE スレッド・プール・アドレス・スペースの数を表します。
- ・「N」は、同時ユーザーの最大数を表します。
- ・「x」は、選択した構成オプションに応じて、以下のいずれかの値になります。

X	SCLMDT	クライアント・ゲートウェイ経由の TSO	APPC 経由の TSO
1	いいえ	いいえ	はい
2	いいえ	はい	いいえ
7	はい	はい	いいえ

- ・「y」は、選択した構成オプションに応じて、以下のいずれかの値になります。

y	
0	CARMA なし
1	CARMA (バッチ)
1	CARMA (crastart)
4	CARMA (ispf)

- ・「z」はデフォルトでは 0 ですが、ユーザーのアクションによっては増える可能性があります。
 - z/OS UNIX シェルが開かれる場合は 1 を加算します。このプロセスは、ユーザーがログオフするまでアクティブです。
 - z/OS UNIX ビルドが実行される場合は 3 を加算します。呼び出されたプログラムの必要性によっては、実際の数値がこれより大きくなる場合があるので注意してください。これらのプロセスは、関連するビルド・タスクが完了した時点で終了します。
 - IBM File Manager との同時対話ごとに 1 を加算します。これらのプロセスは、要求されたオブジェクトが不要になった時点で終了します。
- ・「10 + N*0.05」は、一時プロセス用のバッファを追加します。必要なバッファ・サイズは、ご使用のサイトによって異なる場合があります。

235 ページの図 51 の数式は、Developer for System z クライアントで使用されるプロセスの最大数を見積もる場合に使用します (ここに記載していない一時プロセスはカウントしていません)。

$$(x + y + z) + 5 * s$$

図 51. クライアントごとのプロセス数

各項の説明は次のとおりです。

- 「x」は、選択した構成オプションによって異なります。プロセスの最大数を計算する数式（234 ページの図 50）を参照してください。
- 「y」は、選択した構成オプションによって異なります。プロセスの最大数を計算する数式（234 ページの図 50）を参照してください。
- 「z」はデフォルトでは 0 ですが、ユーザーのアクションによっては増える可能性があります。プロセスの最大数を計算する数式（234 ページの図 50）を参照してください。
- 「s」は、SCLM Developer Toolkit が使用される場合は 1、そうでない場合は 0 です。

表 43 に示す定義によって、プロセスの実際の数を制限することができます。

表 43. プロセスの限度

ロケーション	限度	影響を受けるリソース
BPXPRMxx	MAXPROCSYS	プロセスの総数を制限します。
BPXPRMxx	MAXPROCUSER	z/OS UNIX UID ごとのプロセスの数を制限します。

注:

- RSE デーモンと RSE スレッド・プールは、同じユーザー ID を使用します。RSE デーモンは、必要に応じて新しいスレッド・プールを開始するため、このユーザー ID のプロセス数は増大する可能性があります。そのため、MAXPROCUSER を設定してこの増大に対応する必要があります。これは、「3 + 2*A」と数式化することができます。
- MAXPROCUSER 限度は、固有の z/OS UNIX ユーザー ID (UID) ごとに設定します。複数のユーザーが同じ UID を共用する場合は、見積もったユーザー単位のプロセス数に、同時にアクティブなクライアントの数を乗算します。

スレッドの数

236 ページの表 44 は、選択された Developer for System z の機能で使用するスレッドの数を示しています。「スレッド数」列の「u」は、その機能を使用する同時にアクティブなユーザーの数で値を乗算する必要があることを示します。スレッドの数はプロセス単位で示しています。これは、限度がこのレベルで設定されるためです。

- RSEDx: ここに示すスレッドは、複数のクライアントで共用される RSE スレッド・プールで作成されます。総数を得るには、最終的に同じスレッド・プールに入るすべてのスレッドを合算する必要があります。

- アクティブ: ここに示すスレッドは、要求された機能を実際に実行するプロセスの一部です。各プロセスは独立した単位であるため、特に断りがない限り、それらのプロセスが同じユーザー ID に割り当てられていてもスレッド数を合計する必要はありません。
- ブートストラップ: ブートストラップ・プロセスは、実際のプロセスを開始するために必要です。それぞれが 1 つのスレッドを使用し、複数のブートストラップが連続している可能性があります。スレッド数を合計する必要はありません。

表 44. スレッドの数

スレッド数			ユーザー ID	説明
RSEDx	アクティブ	ブートストラップ		
-	3 + 1u	-	STCJMON	JES ジョブ・モニター
-	10	2	STCLOCK	ロック・デーモン
-	11	2	STCRSE	RSE デーモン
10 (a) + 16u	-	1 (a)	STCRSE	RSE スレッド・プール
-	4u (b)	1u (b)	<userid>	ISPF クライアント・ゲートウェイ (TSO コマンド・サービスおよび SCLMDT)
-	2u	-	<userid>	TSO コマンド・サービス (APPC)
1u	2u	-	STCRSE および <userid>	CARMA (パッチ)
4u	2u	-	STCRSE および <userid>	CARMA (crastart)
5u	4u	3u	STCRSE および <userid>	CARMA (ispf)
-	1u (d)	2u	<userid>	z/OS UNIX ビルド (シェル・コマンド)
6u	1u	-	STCRSE および <userid>	z/OS UNIX シェル
-	2u (c)	-	<userid>	File Manager
-	(5)	-	<userid>	SCLM Developer Toolkit
1u	-	-	STCRSE	アイドル・タイムアウト用のタイマー

注:

- (a) 少なくとも 1 つの RSE スレッド・プール・アドレス・スペースがアクティブです。RSE スレッド・プール・アドレス・スペースの実際の数を確認するには、229 ページの『アドレス・スペースの数』を参照してください。

- (b) 通常の状態ではデフォルトの構成オプションを使用しているときは、ユーザーごとに 1 つの ISPF クライアント・ゲートウェイがアクティブになります。実際数は変わる可能性があります (229 ページの『アドレス・スペースの数』を参照してください)。
- (c) IBM File Manager との対話ごとに 1 つのユーザー固有のプロセスが (示されているスレッド数とともに) 存在します。これらのプロセスは、要求されたオブジェクトが不要になった時点で終了します。
- SCLMDT は、ISPF クライアント・ゲートウェイ・アドレス・スペースを必要とします。SCLMDT は、このアドレス・スペースを TSO コマンド・サービスと共有します。
- 選択されたアクションによっては、SCLMDT がタスク完了時に終了する単一スレッド・プロセスを複数使用することがあります。236 ページの表 44 には、同時 SCLMDT スレッドの最大数を示しています。
- MVS データ・セット関連アクションのほとんどは TSO コマンド・サービスを使用しますが、このサービスは個々に ISPF クライアント・ゲートウェイまたは APPC トランザクションでアクティブとなります。
- (d) z/OS UNIX ビルドがさまざまなビルド・ユーティリティを呼び出して、ビルドがマルチスレッド化することがあります。236 ページの表 44 には、同時 z/OS UNIX ビルド・スレッドの最小数を示しています。
- 特に断りがない限り、リストされているすべてのスレッドは、関連するプロセスが終了するまでアクティブです。

図 52 の数式は、RSE スレッド・プールで使われるスレッドの最大数を見積もる場合に使用します。図 53 の数式は、JES ジョブ・モニターで使われるスレッドの最大数を見積もる場合に使用します。

$$9 + N*(16 + x + y + z) + (20 + N*0.1)$$

図 52. RSE スレッド・プール・スレッドの最大数

$$3 + N$$

図 53. JES ジョブ・モニター・スレッドの最大数

- 各項の説明は次のとおりです。
- 「N」は、このスレッド・プールまたは JES ジョブ・モニターでの同時ユーザーの最大数を表します。デフォルトの設定では、スレッド・プール当たり 60 ユーザーです。
 - 「x」は、選択した構成オプションに応じて、以下のいずれかの値になります。

X	SCLMDT	クライアント・ゲートウェイ経由の TSO	APPC 経由の TSO	タイムアウト
0	いいえ	いいえ	はい	いいえ
0	いいえ	はい	いいえ	いいえ

X	SCLMDT	クライアント・データウェイ経由の TSO	APPC 経由の TSO	タイムアウト
0	はい	はい	いいえ	いいえ
1	いいえ	いいえ	はい	はい
1	いいえ	はい	いいえ	はい
1	はい	はい	いいえ	はい

- 「y」は、選択した構成オプションに応じて、以下のいずれかの値になります。

y	
0	CARMA なし
1	CARMA (パッチ)
4	CARMA (crastart)
5	CARMA (ispf)

- 「z」はデフォルトでは 0 ですが、ユーザーのアクションによっては増える可能性があります。
 - z/OS UNIX シェルが開かれる場合は 6 を加算します。これらのスレッドは、ユーザーがログオフするまでアクティブです。
- 「20 + N*0.1」は、一時スレッド用のバッファを追加します。必要なバッファ・サイズは、ご使用のサイトによって異なる場合があります。

表 45 に示す定義によって、プロセス内の実際のスレッド数を制限することができます。これは、主に RSE スレッド・プールにとって重要です。

表 45. スレッドの限度

ロケーション	限度	影響を受けるリソース
BPXPRMxx	MAXTHREADS	プロセス内のスレッドの数を制限します。
BPXPRMxx	MAXTHREADTASKS	プロセス内の MVS タスクの数を制限します。
BPXPRMxx	MAXASSIZE	アドレス・スペース・サイズを制限し、それによってスレッドに関連する制御ブロックで使用可能なストレージの大きさを制限します。
rsed.envvars	Xmx	Java ヒープ・サイズの最大値を設定します。このストレージは予約され、スレッドに関連する制御ブロックには使用できなくなります。
rsed.envvars	maximum.clients	RSE スレッド・プール内のクライアント (およびそのスレッド) の数を制限します。
rsed.envvars	maximum.threads	RSE スレッド・プール内のクライアント・スレッドの数を制限します。
FEJCNFG	MAX_THREADS	JES ジョブ・モニターでのスレッドの数を制限します。

注: rsed.envvars 内の maximum.threads の値は、BPXPRMxx 内の MAXTHREADS および MAXTHREADTASKS の値よりも小さくする必要があります。

ストレージの使用量

RSE は Java アプリケーションであるため、Developer for System z でのストレージ (メモリー) 使用量を計画する際には、ストレージの割り振りに関する 2 つの限度を考慮に入れる必要があります。それは、Java ヒープ・サイズとアドレス・スペース・サイズです。

Java ヒープ・サイズの限度

Java は、Java アプリケーションのコーディング作業を軽減する多数のサービスを提供しています。そのサービスの 1 つがストレージ管理です。

Java のストレージ管理では、大きなストレージ・ブロックが割り振られ、アプリケーションからの保管要求に使用されます。Java で管理されるこのストレージは、Java ヒープと呼ばれます。定期的なガーベッジ・コレクション (デフラグ) が、ヒープ内の使用されていないスペースをレクラメーション処理して、そのサイズを小さくします。

Java ヒープ・サイズの最大値は、rsed.envvars 内の Xmx ディレクティブで定義されます。このディレクティブが指定されていない場合、Java はデフォルト・サイズの 64 MB を使用します。

各 RSE スレッド・プール (クライアントのアクションをサービスします) は独立した Java アプリケーションであり、そのために専用の Java ヒープを使用します。スレッド・プールはいずれも同じ rsed.envvars 構成ファイルを使用するため、Java ヒープ・サイズの限度が同じであることに注意してください。

スレッド・プールによる Java ヒープの使用量は、接続されているクライアントが実行するアクションによって大きく異なります。最適なヒープ・サイズの限度を設定するには、ヒープの使用量を定期的にモニターすることが必要です。RSE スレッド・プールによる Java ヒープの使用量をモニターするには、**modify display process** オペレーター・コマンドを使用します。

アドレス・スペース・サイズの限度

Java アプリケーションを含むすべての z/OS アプリケーションは、アドレス・スペース内でアクティブとなるため、アドレス・スペース・サイズの限度によって制約を受けます。

必要なアドレス・スペース・サイズは、始動時に JCL の REGION パラメーターなどで指定します。ただし、システム設定によって実際のアドレス・スペース・サイズが制限されることがあります。これらの限度については、162 ページの『アドレス・スペース・サイズ』を参照してください。

- SYS1.PARMLIB(BPXPRMxx) 内の MAXASSIZE
- 開始タスクに割り当てられているユーザー ID の OMVS セグメント内の ASSIZEMAX
- システム出口 IEFUSI および IEALIMIT

RSE スレッド・プールは、RSE デーモンからアドレス・スペース・サイズの限度を継承します。アドレス・スペース・サイズは、Java ヒープ、Java 自体、共通ストレージ域、およびシステムがスレッド・プール・アクティビティをサポートするた

めに作成する全制御ブロック (スレッドごとの TCB (タスク制御ブロック) など) を収容できるだけの大きさであることが必要です。このストレージの使用量の一部は 16 MB 境界より下にあるので注意してください。

アドレス・スペース・サイズに影響する設定 (Java ヒープのサイズや 1つのスレッド・プールでサポートされるユーザー数など) を変更する前に、実際のアドレス・スペース・サイズをモニターする必要があります。Developer for System z による実際のストレージ使用量を追跡するには、通常使用しているシステム・モニター・ソフトウェアを使用します。専用のモニター・ツールがない場合は、SDSF DA ビューや TASID などのツール (ISPF の「Support and downloads」Web ページで入手できる無保証のシステム情報ツール) で基本情報を収集できます。

サイズ見積もりのガイドライン

すでに説明したように、Developer for System z の実際のストレージ使用量は、ユーザーのアクティビティに大きく左右されます。一部のアクションは使用するストレージの量が一定ですが (ログオンなど)、それ以外は変動します (例えば、指定された高位修飾子を持つデータ・セットをリストする処理など)。

- RSE には、Java ヒープとすべてのシステム制御ブロックを収容できるように、2 GB のアドレス・スペースを使用してください。
- サンプルの `rsed.envvars` セットアップでは、スレッド・プール当たりのユーザー数が 60 に設定されています。
 - `maximum.clients=60`
 - `maximum.threads=1000` ($10+16*60 = 970$ 、したがって 1000 の場合のクライアント数は 61)
- サンプルの `rsed.envvars` セットアップでは、Java ヒープを 256 MB まで拡大できるようになっています。これにより、60 のクライアントそれぞれが平均で 4 MB を使用できます ($60*4 = 240$)。

RSE の始動時に、コンソール・メッセージ FEK004I で、現行の Java ヒープおよびアドレス・スペース・サイズの限度が表示されるので注意してください。

モニター中に現行の Java ヒープ・サイズが実際のワークロードに対して不十分であることがわかった場合は、以下のいずれかのシナリオを使用します。

- `rsed.envvars` の `Xmx` ディレクティブで、Java ヒープ・サイズの最大値を増やします。これを行う前に、アドレス・スペースにサイズを増やす余地があることを確認してください。
- `rsed.envvars` の `maximum.clients` ディレクティブで、スレッド・プール当たりのクライアントの最大数を減らします。RSE がサポートするクライアントの数は変わりませんが、クライアントが分散されるスレッド・プールの数が増えます。

ストレージ使用量分析のサンプル

以下の各図には、1 箇所を変更したデフォルトの Developer for System z セットアップに関するサンプルのリソース使用量の数値が示されています。Java ヒープ・サイズの最大値が 10 MB に設定されています。これは、最大値を小さくすることで使用率が高くなり、ヒープ・サイズの限度により早く到達するためです。

Max Heap Size=10MB and private AS Size=1,959MB

startup

BPXM023I (STCRSE)
ProcessId(268) Memory Usage(7%) Clients(0)

Jobname	Cpu time	Storage	EXCP
JMON	0.01	2740	72
LOCKD	1.60	28.7M	14183
RSED	4.47	32.8M	15910
RSED8	1.15	27.4M	12612

logon 1

BPXM023I (STCRSE)
ProcessId(268) Memory Usage(13%) Clients(1)

Jobname	Cpu time	Storage	EXCP
JMON	0.01	2864	81
LOCKD	1.64	28.8M	14259
RSED	4.55	32.8M	15980
RSED8	3.72	55.9M	24128

logon 2

BPXM023I (STCRSE)
ProcessId(268) Memory Usage(23%) Clients(2)

Jobname	Cpu time	Storage	EXCP
JMON	0.02	2944	86
LOCKD	1.66	28.9M	14268
RSED	4.58	32.9M	16027
RSED8	4.20	57.8M	25205

logon 3

BPXM023I (STCRSE)
ProcessId(268) Memory Usage(37%) Clients(3)

Jobname	Cpu time	Storage	EXCP
JMON	0.02	3020	91
LOCKD	1.67	29.0M	14277
RSED	4.60	32.9M	16076
RSED8	4.51	59.6M	26327

logon 4

BPXM023I (STCRSE)
ProcessId(268) Memory Usage(41%) Clients(4)

Jobname	Cpu time	Storage	EXCP
JMON	0.02	3108	96
LOCKD	1.68	29.0M	14286
RSED	4.61	32.9M	16125
RSED8	4.77	62.3M	27404

図 54. ログオン数 5 の場合のリソース使用量

logon 5

```
BPXM023I (STCRSE)
ProcessId(268      ) Memory Usage(41%) Clients(4)
ProcessId(33554706) Memory Usage(13%) Clients(1)
```

Jobname	Cpu time	Storage	EXCP
JMON	0.03	3184	101
LOCKD	1.69	29.1M	14295
RSED	4.64	32.9M	16229
RSED8	4.78	62.4M	27413
RSED9	4.60	56.6M	24065

図 55. ログオン数 5 の場合のリソース使用量 (続き)

241 ページの図 54 と 図 55 は、Java ヒープが 10 MB に設定された RSE デーモンに 5 つのクライアントがログオンしているシナリオを示しています。

- スレッド・プール (RSED8) は、開始された時点で休止状態にあり、約 27 MB を使用しています。そのうち 0.7 MB は Java ヒープ内にあります (10 MB の 7%)。
- このスレッド・プールは、最初のクライアントが接続するとアクティブになって新たに 27 MB を使用し、クライアントが接続するたびに追加で 2 MB ずつ使用します。
- ヒープ使用量が増えていることからわかるように、この接続当たりの 2 MB の一部は Java ヒープ内にあります。
- ただし、ヒープ使用量は、必要なストレージを見積もって必要以上に割り振る Java のメカニズムに依存するため、実際のパターンは存在しません。断続的なガーベッジ・コレクションによってストレージが解放されるため、傾向を見つけ出すことはさらに難しくなります。
- 内部のメカニズムによって、アクティブなスレッドに十分なヒープ・サイズを確保できるようにスレッド・プール当たりの接続数が制限されているため、5 番目の接続は新しいスレッド・プール (RSED9) に作成されます。正しく構成されたセットアップを使用している場合は、他の限度に先に到達するため (最も可能性が高いのは `rsed.envvars` 内の `maximum.clients`)、通常はこのような内部の安全策が実行されることはありません。

Max Heap Size=10MB and private AS Size=1,959MB

startup

BPXM023I (STCRSE)
ProcessId(212) Memory Usage(7%) Clients(0)

Jobname	Cpu time	Storage	EXCP
JMON	0.01	2736	71
LOCKD	1.73	30.5M	14179
RSED	4.35	32.9M	15117
RSED8	1.43	27.4M	12609

logon

BPXM023I (STCRSE)
ProcessId(212) Memory Usage(13%) Clients(1)

Jobname	Cpu time	Storage	EXCP
JMON	0.01	2864	80
LOCKD	1.76	30.6M	14255
RSED	4.48	33.0M	15187
RSED8	3.53	53.9M	24125

expand large MVS tree (195 data sets)

BPXM023I (STCRSE)
ProcessId(212) Memory Usage(13%) Clients(1)

Jobname	Cpu time	Storage	EXCP
JMON	0.01	2864	80
LOCKD	1.78	30.6M	14255
RSED	4.58	33.1M	16094
RSED8	4.28	56.1M	24740

expand small PDS (21 members)

BPXM023I (STCRSE)
ProcessId(212) Memory Usage(13%) Clients(1)

Jobname	Cpu time	Storage	EXCP
IBMUSER2	0.22	2644	870
JMON	0.01	2864	80
LOCKD	1.78	30.6M	14255
RSED	4.61	33.1M	16108
RSED8	4.40	56.2M	24937

open medium sized member (86 lines)

BPXM023I (STCRSE)
ProcessId(212) Memory Usage(13%) Clients(1)

Jobname	Cpu time	Storage	EXCP
IBMUSER2	0.22	2644	870
JMON	0.01	2864	80
RSED	4.61	33.1M	16108
RSED8	8.12	62.7M	27044

図 56. PDS メンバー編集時のリソース使用量

図 56 は、Java ヒープが 10 MB の RSE デーモンに 1 つのクライアントがログオンし、PDS メンバーを編集するシナリオを示しています。

- 195 のデータ・セット名を検出したカタログ検索で約 2 MB のストレージが使用されています。Java ヒープ使用量は増えていないことから、これはすべてシステム・アクティビティーによるものです。
- メンバー数 21 の PDS をオープンする操作ではスレッド・プール内のメモリーはほとんど使用されませんが、表示内容から TSO コマンド・サービスが呼び出されたことがわかります。新しいアドレス・スペース (IBMUUSER2) がアクティブになり、TSO でこのユーザーに割り当てられている領域サイズを使用します。このアドレス・スペースは指定された時間内はアクティブであるため、TSO コマンド・サービスによるその後の要求に再利用できます。
- メンバーをオープンすると、高位修飾子を拡張するときと同じような数値が示されます。Java ヒープ使用量は変わりませんが、システム・アクティビティーのためにストレージが 6.5 MB 増えています。

z/OS UNIX ファイル・システム・スペースの使用量

DD ステートメントに書き込まれない Developer for System z 関連データのほとんどは、最終的に z/OS UNIX ファイルに格納されます。システム・プログラマーは、どのデータが書き込まれ、どこに格納されるかを制御できます。ただし、書き込まれるデータの量を制御することはできません。

データは、次のカテゴリーに分類することができます。

- 問題分析 (ログ・ファイルとシステム・ダンプ・ファイル)。これについては、145 ページの『第 9 章 構成問題のトラブルシューティング』で詳しく説明しています。
- 監査。これについては 176 ページの『監査ロギング』で説明しています。
- 一時データ

145 ページの『第 9 章 構成問題のトラブルシューティング』で説明したように、Developer for System z は、RSE 関連のホスト・ログを以下の z/OS UNIX ディレクトリーに書き込みます。

- /var/rdz/logs (RSE 開始タスク・ログ)
- /var/rdz/logs/\$LOGNAME (ユーザー・ログ)

デフォルトでは、エラー・メッセージと警告メッセージだけがログに書き込まれます。そのため、すべてが計画したとおりに進めば、上記のディレクトリーにはほとんど、またはまったくファイルが存在しないはず (監査ログは対象外です)。

情報メッセージのロギングを有効にすることができますが (本来は IBM サポートの指示の下で行います)、ログ・ファイルのサイズが著しく増大します。

```

startup

$ ls -l /var/rdz/logs
total 144
-rw-rw-rw- 1 STCRSE STCGRP 33642 Jul 10 12:10 rsedaemon.log
-rw-rw-rw- 1 STCRSE STCGRP 1442 Jul 10 12:10 rseserver.log

logon

$ ls -l /var/rdz/logs
total 144
drwxrwxrwx 3 IBMUSER SYS1 8192 Jul 10 12:11 IBMUSER
-rw-rw-rw- 1 STCRSE STCGRP 36655 Jul 10 12:11 rsedaemon.log
-rw-rw-rw- 1 STCRSE STCGRP 1893 Jul 10 12:11 rseserver.log
$ ls -l /var/rdz/logs/IBMUSER
total 160
-rw-rw-rw- 1 IBMUSER SYS1 3459 Jul 10 12:11 ffs.log
-rw-rw-rw- 1 IBMUSER SYS1 0 Jul 10 12:11 ffsget.log
-rw-rw-rw- 1 IBMUSER SYS1 0 Jul 10 12:11 ffsput.log
-rw-rw-rw- 1 IBMUSER SYS1 303 Jul 10 12:11 lock.log
-rw-rw-rw- 1 IBMUSER SYS1 126 Jul 10 12:11 rmt_classloader_cache.jar
-rw-rw-rw- 1 IBMUSER SYS1 7266 Jul 10 12:11 rsecomm.log
-rw-rw-rw- 1 IBMUSER SYS1 0 Jul 10 12:11 stderr.log
-rw-rw-rw- 1 IBMUSER SYS1 0 Jul 10 12:11 stdout.log

logoff

$ ls -l /var/rdz/logs
total 80
drwxrwxrwx 3 IBMUSER SYS1 8192 Jul 10 12:11 IBMUSER
-rw-rw-rw- 1 STCRSE STCGRP 36655 Jul 10 12:11 rsedaemon.log
-rw-rw-rw- 1 STCRSE STCGRP 2208 Jul 10 12:11 rseserver.log
$ ls -l /var/rdz/logs/IBMUSER
total 296
-rw-rw-rw- 1 IBMUSER SYS1 6393 Jul 10 12:11 ffs.log
-rw-rw-rw- 1 IBMUSER SYS1 0 Jul 10 12:11 ffsget.log
-rw-rw-rw- 1 IBMUSER SYS1 0 Jul 10 12:11 ffsput.log
-rw-rw-rw- 1 IBMUSER SYS1 609 Jul 10 12:11 lock.log
-rw-rw-rw- 1 IBMUSER SYS1 126 Jul 10 12:11 rmt_classloader_cache.jar
-rw-rw-rw- 1 IBMUSER SYS1 45157 Jul 10 12:11 rsecomm.log
-rw-rw-rw- 1 IBMUSER SYS1 0 Jul 10 12:11 stderr.log
-rw-rw-rw- 1 IBMUSER SYS1 176 Jul 10 12:11 stdout.log

stop

$ ls -l /var/rdz/logs
total 80
drwxrwxrwx 3 IBMUSER SYS1 8192 Jul 10 12:11 IBMUSER
-rw-rw-rw- 1 STCRSE STCGRP 36655 Jul 10 12:11 rsedaemon.log
-rw-rw-rw- 1 STCRSE STCGRP 2490 Jul 10 12:12 rseserver.log

```

図 57. z/OS UNIX ファイル・システム・スペースの使用量

図 57 は、デバッグ・レベル 2 (情報メッセージ) を使用する場合は z/OS UNIX ファイル・システム・スペースの最小使用量を示しています。

- 開始タスク・ログは開始後に 34 KB を使用し、ユーザーのログオン、ログオフ、またはオペレーター・コマンドの発行に伴って少しずつ増大します。
- クライアント・ログ・ディレクトリーはログオン後に 11 KB を使用し、ユーザーが作業を開始すると一定のペースで増大します (サンプルには示されていません)。
- ログオフによって新たに 40 KB がユーザー・ログに追加されて、51 KB になります。

監査ログ以外のログ・ファイルは、再始動 (RSE 開始タスクの場合) またはログオン (クライアントの場合) のたびに上書きされて、合計サイズを抑えます。この動作は、rsed.envvars 内の keep.last.log ディレクティブによって多少変わります。このディレクティブでは、前のログのコピーを残すように RSE に指示できるからです。それより古いコピーは必ず除去されます。

監査がアクティブであるときに、監査ログ・ファイルを保持しているファイル・システムのフリー・スペースが不足してくると、警告メッセージがコンソールに送信されます。このコンソール・メッセージ (FEK103E) は、スペース不足の問題が解決されるまで定期的に繰り返されます。RSE が生成するコンソール・メッセージのリストについては、140 ページの『コンソール・メッセージ』を参照してください。

表 46 に示す定義は、ログ・ディレクトリーに書き込まれるデータおよびディレクトリーのロケーションを制御します。

表 46. ログ出力ディレクティブ

ロケーション	ディレクティブ	機能
resecomm.properties	debug_level	デフォルトのログ詳細レベルを設定します。
rsed.envvars	keep.last.log	開始/ログオン前に、前のログのコピーを保存します。
rsed.envvars	enable.audit.log	クライアント・アクションの監査トレースを保存します。
rsed.envvars	enable.standard.log	スレッド・プール (1 つまたは複数) の stdout および stderr ストリームをログ・ファイルに書き込みます。
rsed.envvars	DSTORE_TRACING_ON	DataStore アクションのロギングを有効にします。
rsed.envvars	DSTORE_MEMLOGGING_ON	DataStore メモリ使用量のロギングを有効にします。
オペレーター・コマンド	modify rsecommlog <level>	rsecomm.log のログ詳細レベルを動的に変更します。
オペレーター・コマンド	modify rsedaemonlog <level>	rsedaemon.log のログ詳細レベルを動的に変更します。
オペレーター・コマンド	modify rseserverlog <level>	rseserver.log のログ詳細レベルを動的に変更します。
オペレーター・コマンド	modify rsestandardlog {on/off}	std*.*.log の更新を動的に変更します。
rsed.envvars	daemon.log	RSE 開始タスク・ログおよび監査ログのホーム・パス。
rsed.envvars	user.log	ユーザー・ログのホーム・パス

これに加えて、Developer for System z は、ISPF クライアント・ゲートウェイなどの必要なソフトウェアとともに、一時データを /tmp および /var/rdz/WORKAREA に書き込みます。ユーザー・アクションの結果としてここに書き込まれるデータの量は予測不能であるため、これらのディレクトリーを保持するファイル・システムに十分なフリー・スペースを確保しておく必要があります。

Developer for System z は、これらの一時ファイルのクリーンアップを常時試みますが、115 ページの『(オプション) WORKAREA クリーンアップ』で説明するように、手動でのクリーンアップも随時実行できます。

主要なリソース定義

/etc/rdz/rsed.envvars

rsed.envvars で定義されている環境変数は、RSE、Java、および z/OS UNIX によって使用されます。Developer for System z 付属のサンプル・ファイルは、Developer for System z のオプション・コンポーネントを必要としない小規模から中規模のインストール済み環境を対象としています。サンプル・ファイルに定義されている各変数については 35 ページの『rsed.envvars、RSE 構成ファイル』で説明していますが、以下の変数には特に注意が必要です。

`_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Xms128m -Xmx256m"`

初期 (Xms) および最大 (Xmx) ヒープ・サイズを設定します。デフォルトはそれぞれ、128M と 256M です。希望するヒープ・サイズ値を強制的に使用させるには、変更してください。このディレクティブがコメント化されている場合は、Java のデフォルト値が使用されます。デフォルト値はそれぞれ 4M と 512M です (Java 5.0 の場合は 1M と 64M)。

`#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dmaximum.clients=60"`

1 つのスレッド・プールでサービスできるクライアントの最大数。デフォルトは 60 です。1 スレッド・プール当たりのクライアント数を制限するには、コメント解除してカスタマイズします。他の制限のために、RSE がこの限度に到達しない場合もあることに注意してください。

`#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dmaximum.threads=1000"`

新規クライアントを許可するための、1 つのスレッド・プールでのアクティブ・スレッド数の最大値。デフォルトは 1000 です。1 スレッド・プール当たりのクライアント数を、使用中のスレッド数に基づいて制限するには、コメント解除してカスタマイズします。それぞれのクライアント接続が複数 (16 以上) のスレッドを使用すること、および他の制限のために RSE がこの限度に到達しない場合があることに注意してください。

注: この値は、SYS1.PARMLIB(BPXPRMxx) での MAXTHREADS および MAXTHREADTASKS の設定より小さくする必要があります。

`#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dminimum.threadpool.process=10"`

アクティブ・スレッド・プールの最小数。デフォルトは 1 です。少なくともリストされた数のスレッド・プール・プロセスを開始するには、コメント解除してカスタマイズします。スレッド・プール・プロセスは、RSE サーバー・スレッドのロード・バランシングに使用されます。より多くの新規プロセスが必要になった場合は、その時点で新規プロセスが開始されます。前もって新規プロセスを開始しておく、接続遅延を回避できますが、アイドル時間中に使用されるリソースが増えます。

`#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Dmaximum.threadpool.process=100"`

アクティブ・スレッド・プールの最大数。デフォルトは 100 です。スレッド・プール・プロセスの数を制限するには、コメント解除してカスタマイズします。

スレッド・プール・プロセスは、RSE サーバー・スレッドのロード・バランシングに使用されます。このため、スレッドを制限すると、アクティブなクライアント接続の数も制限されます。

SYS1.PARMLIB(BPXPRMxx)

RSE は Java アプリケーションであるため、z/OS UNIX 環境でアクティブになります。このため、z/OS UNIX の環境とファイル・システムを制御するパラメーターを含んでいる BPXPRMxx は、非常に重要な parmlib メンバーとなります。BPXPRMxx については、「MVS 初期設定およびチューニング解説書」(SA88-8564)で説明されています。Developer for System z に影響することがわかっているディレクティブは以下のとおりです。

MAXPROCSYS(nnnnn)

システムで許可されるプロセスの最大数を指定します。

値の範囲: nnnnn は 5 から 32767 までの 10 進値です。
デフォルト: 900

MAXPROCUSER(nnnnn)

プロセスが作成された方法に関係なく、単一の z/OS UNIX ユーザー ID が同時にアクティブにしておくことができるプロセスの最大数を指定します。

値の範囲: nnnnn は 3 から 32767 までの 10 進値です。
デフォルト: 25

注:

- RSE プロセスはいずれも同じ z/OS UNIX ユーザー ID (RSE デーモンに割り当てられているユーザーの ID) を使用します。これは、すべてのクライアントが RSE プロセス内のスレッドとして実行されるためです。
- この値は、RSED 開始タスクに割り当てられているユーザーの OMVS セキュリティー・プロファイル・セグメント内の PROCUSERMAX 変数で設定することもできます。

MAXTHREADS(nnnnnn)

単一のプロセスが同時にアクティブにしておくことができる pthread_created スレッドの最大数を指定します。これには、実行中のスレッド、キューに入れられたスレッド、および終了してまだ切り離されていないスレッドが含まれます。値 0 を指定すると、アプリケーションで pthread_create が使用されなくなります。

値の範囲: nnnnnn は 0 から 100000 までの 10 進値です。
デフォルト: 200

注:

- 各クライアントは、RSE スレッド・プール・プロセス内で少なくとも 16 のスレッドを使用し、プロセス内では複数のクライアントがアクティブになります。
- この値は、RSED 開始タスクに割り当てられているユーザーの OMVS セキュリティー・プロファイル・セグメント内の THREADSMAX 変数で設定することもできます。設定された THREADSMAX 値は、MAXTHREADS と MAXTHREADTASKS の両方に使用されます。

MAXTHREADTASKS(nnnnnn)

単一プロセスが pthread_created スレッドに対して同時にアクティブにしておくことができる MVS タスクの最大数を指定します。

値の範囲: nnnnnn は 0 から 32768 までの 10 進値です。

デフォルト: 1000

注:

- アクティブなスレッドそれぞれに MVS タスク (TCB、タスク制御ブロック) があります。
- 同時 MVS タスクそれぞれに追加のストレージが必要で、その一部は 16 MB 境界より下に位置している必要があります。
- 各クライアントは、RSE スレッド・プール・プロセス内で少なくとも 16 のスレッドを使用し、プロセス内では複数のクライアントがアクティブになります。
- この値は、RSED 開始タスクに割り当てられているユーザーの OMVS セキュリティー・プロファイル・セグメント内の THREADSMAX 変数で設定することもできます。設定された THREADSMAX 値は、MAXTHREADS と MAXTHREADTASKS の両方に使用されます。

MAXUIDS(nnnnnn)

同時に操作を実行できる z/OS UNIX ユーザー ID (UID) の最大数を指定します。

値の範囲: nnnnnn は 1 から 32767 までの 10 進値です。

デフォルト: 200

MAXASSIZE(nnnnnn)

新しいプロセスの初期値として設定される RLIMIT_AS リソース値を指定します。RLIMIT_AS は、アドレス・スペースの領域サイズを示します。

値の範囲: nnnnnn は 10485760 (10 MB) から 2147483647 (2 GB) までの 10 進値です。デフォルト: 209715200 (200 MB)

注:

- この値は 2G に設定する必要があります。
- この値は、RSED 開始タスクに割り当てられているユーザーの OMVS セキュリティー・プロファイル・セグメント内の ASSIZEMAX 変数で設定することもできます。

MAXFILEPROC(nnnnnnn)

単一プロセスが同時にアクティブにするか、割り振ることができる、ファイル、ソケット、ディレクトリー、およびその他のファイル・システム・オブジェクトの記述子の最大数を指定します。

値の範囲: nnnnnnn は 3 から 524287 までの 10 進値です。

デフォルト: 64000

注:

- スレッド・プールでは、そのすべてのクライアント・スレッドが単一のプロセスに入っています。

- この値は、RSED 開始タスクに割り当てられているユーザーの OMVS セキュリティー・プロファイル・セグメント内の FILEPROCMAX 変数で設定することもできます。

MAXMMAPAREA(nnnnnn)

z/OS UNIX ファイルのメモリー・マッピングに割り振ることのできるデータ・スペース・ストレージ・スペースの最大量 (ページ単位) を指定します。メモリー・マッピングがアクティブになるまで、ストレージは割り振られません。

値の範囲: nnnnn は 1 から 16777216 までの 10 進値です。
デフォルト: 40960

注: この値は、RSED 開始タスクに割り当てられているユーザーの OMVS セキュリティー・プロファイル・セグメント内の MMAPAREAMAX 変数で設定することもできます。

元の BPXPRMxx 変数の値を動的に (次の IPL まで) 増減させるには、オペレーター・コマンド **SETOMVS** または **SET OMVS** を使用します。永続的な変更を加える場合は、IPL に使用される BPXPRMxx メンバーを編集します。これらのオペレーター・コマンドの詳細については、「MVS システム・コマンド」(SA88-8593) を参照してください。

以下の定義は、NETWORK ステートメントのサブパラメーターです。

MAXSOCKETS(nnnnnnnn)

このアドレス・ファミリーに対してこのファイル・システムでサポートされる最大ソケット数を指定します。これはオプション・パラメーターです。

値の範囲: nnnnnnnn は 0 から 16777215 までの 10 進値です。
デフォルト: 100

INADDRANYCOUNT(nnnn)

システムが PORT 0、INADDR_ANY バインドに使用するために予約する、INADDRANYPORT パラメーターで指定したポート番号から始まるポートの数を指定します。この値は、CINET (複数の TCP/IP スタック) にのみ必要です。

値の範囲: nnnn は 1 から 4000 までの 10 進値です。
デフォルト: INADDRANYPORT と INADDRANYCOUNT がどちらも指定されていない場合、INADDRANYCOUNT のデフォルトは 1000 です。
それ以外の場合はポートが予約されません (0)。

さまざまなリソース定義

サーバー JCL での EXEC カード

以下の定義は、Developer for System z サーバーの JCL の EXEC カードに追加することをお勧めします。

REGION=0M

RSE デーモンおよび JES ジョブ・モニターの開始タスク (それぞれ RSED と JMON) には REGION=0M を指定することをお勧めします。そうすることで、ア

ドレス・スペース・サイズが、使用可能な専用ストレージ、あるいは IEFUSI または IEALIMIT システム出口によってのみ制限されます。IBM では、z/OS UNIX アドレス・スペースには RSE デーモンのようにこれらの出口を使用しないよう強く推奨しています。

TIME=NOLIMIT

すべての Developer for System z サーバーに TIME=NOLIMIT を使用することをお勧めします。これは、すべての Developer for System z クライアントの CPU 時間がサーバー・アドレス・スペース内で集計されるためです。

FEK.#CUST.PARMLIB(FEJJCNFG)

FEJJCNFG で定義されている環境変数は、JES ジョブ・モニターで使用されます。Developer for System z 付属のこのサンプル・ファイルは、小規模から中規模のインストール済み環境を対象としています。サンプル・ファイルに定義されている各変数については、30 ページの『FEJJCNFG、JES ジョブ・モニター構成ファイル』で説明していますが、以下の変数には特に注意が必要です。

MAX_THREADS

1 つの JES ジョブ・モニターを一度に使用できるユーザーの最大数。デフォルトは 200 です。最大値は 2147483647 です。この数を大きくすると、JES ジョブ・モニターのアドレス・スペースのサイズを大きくしなければならない場合があります。

SYS1.PARMLIB(IEASYSxx)

IEASYSxx はシステム・パラメーターを保持するもので、「MVS 初期設定およびチューニング解説書」(SA88-8564) に説明があります。Developer for System z に影響することがわかっているディレクティブは以下のとおりです。

MAXUSER=nnnnn

このパラメーターは、ほとんどの条件下で、特定の IPL 時に同時に実行できるジョブおよび開始タスクの数を制限するためにシステムが使用する値を指定します。

値の範囲: nnnnn は 0 から 32767 までの 10 進値です。

MAXUSER、RSVSTRT、および RSVNONR の各システム・パラメーターに指定された値の合計は 32767 以下でなければならないので注意してください。

デフォルト値: 255

SYS1.PARMLIB(IVTPRMxx)

IVTPRMxx は通信ストレージ・マネージャー (CSM) のパラメーターを設定するもので、「MVS 初期設定およびチューニング解説書」(SA88-8564) に説明があります。Developer for System z に影響することがわかっているディレクティブは以下のとおりです。

FIXED MAX(maxfix)

固定 CSM バッファ専用ストレージの最大量を定義します。

値の範囲: 1024K から 2048M までの値。

デフォルト: 100M

ECSA MAX(maxecsa)

ECSA CSM バッファー専用のストレージの最大量を定義します。

値の範囲: 1024K から 2048M までの値。

デフォルト: 100M

SYS1.PARMLIB(ASCHPMxx)

ASCHPMxx parmlib メンバーは、ASCH トランザクション・スケジューラーのスケジューリング情報を格納するもので、「MVS 初期設定およびチューニング解説書」(SA88-8564) に説明があります。Developer for System z に影響することがわかっているディレクティブは以下のとおりです。

MAX(nnnnn)

CLASSADD 定義のオプション・パラメーター。トランザクション・イニシエーターの特定のクラスに対して許可される APPC トランザクション・イニシエーターの最大数を指定します。この限度に到達すると、新しいアドレス・スペースが作成されなくなり、着信要求がキューに入れられて、既存のイニシエーター・アドレス・スペースが使用可能になるまで待機します。この値は、ご使用のシステムで許可されている最大アドレス・スペース数を超えないようにする必要があります。また、同じくアドレス・スペースを必要とするシステム上の競合製品に留意してください。

値の範囲: nnnnn は 1 から 64000 までの 10 進値です。

デフォルト: 1

注: APPC を使用して TSO コマンド・サービスを開始する場合は、使用されるトランザクション・クラスに、Developer for System z の各同時ユーザーに 1 つずつイニシエーターを許可できるだけの十分な数のトランザクション・イニシエーターがあることが必要です。

モニター

ユーザーのワークロードによってシステム・リソースの必要性が変わる可能性があるため、ユーザーの要件に応じて Rational Developer for System z とシステムの構成を調整できるように、定期的にシステムをモニターしてリソース使用量を測定する必要があります。このモニター処理に役立つコマンドを以下に示します。

RSE のモニター

RSE スレッド・プールは、Developer for System z におけるユーザー・アクティビティのフォーカル・ポイントであるため、最適に利用されているかモニターする必要があります。通常のシステム・モニター・ツールでは収集できない情報は、RSE デーモンに照会することができます。

- アドレス・スペース固有のデータ (使用されている実ストレージや CPU 時間など) を収集するには、RMF などの通常のシステム・モニター・ツールを使用します。専用のモニター・ツールがない場合は、SDSF DA ビューや TASID などのツール (ISPF の「Support and downloads」Web ページで入手できる無保証のシステム情報ツール) で基本情報を収集できます。

- RSE デーモンは、使用可能なアドレス・スペース・サイズと Java ヒープ・サイズを、始動時にコンソール・メッセージ FEK004I で通知します。

```
FEK004I RseDaemon: Max Heap Size=65MB and private AS Size=1,959MB
```

- **MODIFY RSED,APPL=DISPLAY PROCESS** オペレーター・コマンドは、RSE スレッド・プール・プロセスを表示します。「Memory Usage」フィールドには、定義された Java ヒープが実際にどの程度使用されているかが示されます。このコマンドの詳細については、133 ページの『第 8 章 オペレーター・コマンド』を参照してください。

```
f rsed,appl=d p
BPXM023I (STCRSE)
ProcessId(16777456) Memory Usage(33%) Clients(4) Order(1)
```

DISPLAY PROCESS 変更コマンドの DETAIL オプションを使用すると、詳細情報が表示されます。

```
f rsed,appl=d p,detail
BPXM023I (STCRSE)
ProcessId(33555087) ASId(002E) JobName(RSED8) Order(1)
PROCESS LIMITS:      CURRENT  HIGHWATER    LIMIT
JAVA HEAP USAGE(%)   10       56          100
CLIENTS               0       25           60
MAXFILEPROC           83      103        64000
MAXPROCUSER           97       99          200
MAXTHREADS            9       14         1500
MAXTHREADTASKS        9       14         1500
```

z/OS UNIX のモニター

Developer for System z に関する z/OS UNIX の限度のほとんどは、オペレーター・コマンドを使用して表示できます。一部のコマンドでは、特定の限度に関する現在の使用率と最高水準点も示されます。これらのコマンドの詳細については、「MVS システム・コマンド」(SA88-8593) を参照してください。

- SYS1.PARMLIB(BPXPRMxx) 内の LIMMSG(ALL) ディレクティブは、parmlib のいずれかの限度に到達しそうなときにコンソール・メッセージ (BPXI040I) を表示するよう z/OS UNIX に指示します。LIMMSG のデフォルト値は NONE です。この値が指定されると機能は無効になります。この機能を動的にアクティブにする (次の IPL まで) には、オペレーター・コマンド **SETOMVS LIMMSG=ALL** を使用します。このディレクティブの詳細については、「MVS 初期設定およびチューニング解説書」(SA88-8564) を参照してください。
- **DISPLAY OMVS,OPTIONS** オペレーター・コマンドは、動的に設定できる z/OS UNIX ディレクティブの現行値を表示します。

```
d omvs,o
BPX0043I 13.10.16 DISPLAY OMVS 066
OMVS      000D ETC/INIT WAIT  OMVS=(M7)
CURRENT UNIX CONFIGURATION SETTINGS:
MAXPROCSYS      =      256    MAXPROCUSER      =      16
MAXFILEPROC     =      256    MAXFILESIZE       = NOLIMIT
MAXCPUPTIME     =     1000    MAXUIDS        =      200
MAXPTYS         =      256
MAXMMAPAREA     =      256    MAXASSIZE       = 209715200
MAXTHREADS      =      200    MAXTHREADTASKS =      1000
MAXCORESIZE     =    4194304  MAXSHAREPAGES =      4096
IPCMSGQBYTES    = 2147483647  IPCMSGQMNUM   =     10000
IPCMSGNIDS      =      500    IPCSEMNIDS    =      500
IPCSEMNOPS      =      25     IPCSEMNSEMS    =     1000
IPCshmPAGES     =    25600    IPCSHMNIDS    =      500
```

```

IPCSHMNSEGS      =          500      IPCSHMSPAGES      =      262144
SUPERUSER        = BPXROOT          FORKCOPY          = COW
STEPLIBLIST      =
USERIDALIASTABLE=
SERV_LINKLIB     = POSIX.DYN SERV.LOADLIB  BPXLK1
SERV_LPALIB      = POSIX.DYN SERV.LOADLIB  BPXLK1
PRIORITYPG VALUES: NONE
PRIORITYGOAL VALUES: NONE
MAXQUEUEDSIGs    =          1000      SHRLIBRGNSIZE    =      67108864
SHRLIBMAXPAGES   =          4096      VERSION           = /
SYSCALL COUNTS   = NO                 TTYGROUP        = TTY
SYSPLEX          = NO                 BRM SERVER      = N/A
LIMMSG           = NONE               AUTOCVT         = OFF
RESOLVER PROC    = DEFAULT
AUTHPGMLIST      = NONE
SWA              = BELOW

```

- **DISPLAY OMVS,LIMITS** オペレーター・コマンドは、現在設定されている z/OS UNIX システム・サービスの parmlib 限度、それらの最高水準点、および現行のシステム使用量に関する情報を表示します。

```

d omvs,l
BPX0051I 14.05.52 DISPLAY OMVS 904
OMVS      0042 ACTIVE              OMVS=(69)
SYSTEM WIDE LIMITS:                LIMMSG=SYSTEM

```

	CURRENT USAGE	HIGHWATER USAGE	SYSTEM LIMIT
MAXPROCSYS	1	4	256
MAXUIDS	0	0	200
MAXPTYS	0	0	256
MAXMMAPAREA	0	0	256
MAXSHAREPAGES	0	10	4096
IPCMsGNIDS	0	0	500
IPCSEMNIDS	0	0	500
IPCSHMNIDS	0	0	500
IPCSHMSPAGES	0	0	262144 *
IPCMSGQBYTES	---	0	262144
IPCMSGQNUM	---	0	10000
IPCSHMMPAGES	---	0	256
SHRLIBRGNSIZE	0	0	67108864
SHRLIBMAXPAGES	0	0	4096

このコマンドで PID=processid キーワードを指定すると、個々のプロセスの最高水準点と現行の使用量が表示されます。

```

d,omvs,l,pid=16777456
BPX0051I 14.06.28 DISPLAY OMVS 645
OMVS      000E ACTIVE              OMVS=(76)
USER      JOBNAME  ASID      PID      PPID STATE  START  CT_SECS
STRCE     RSED8    007E     16777456  67109106 HF---- 20.00.56 113.914
LATCHWAITPID= 0 CMD=java -Ddaemon.log=/var/rdz/logs -
PROCESS LIMITS:                LIMMSG=NONE

```

	CURRENT USAGE	HIGHWATER USAGE	PROCESS LIMIT
MAXFILEPROC	83	103	256
MAXFILESIZE	---	---	NOLIMIT
MAXPROCUSER	97	99	200
MAXQUEUEDSIGs	0	1	1000
MAXTHREADS	9	14	200
MAXTHREADTASKS	9	14	1000
IPCSHMNSEGS	0	0	500
MAXCORESIZE	---	---	4194304
MAXMEMLIMIT	0	0	16383P

- **DISPLAY OMVS,PFS** オペレーター・コマンドは、z/OS UNIX 構成に現在組み込まれている各物理ファイル・システム (TCP/IP スタックを含む) に関する情報を表示します。

```
d omvs,p
BPX0046I 14.35.38 DISPLAY OMVS 092
OMVS 000E ACTIVE OMVS=(33)
PFS CONFIGURATION INFORMATION
PFS TYPE DESCRIPTION ENTRY MAXSOCK OPNSOCK HIGHUSED
TCP SOCKETS AF_INET EZBPFINI 50000 244 8146
UDS SOCKETS AF_UNIX BPXTUINI 64 6 10
ZFS LOCAL FILE SYSTEM IOEFSCM
14:32.00 RECYCLING
HFS LOCAL FILE SYSTEM GFUAINIT
BPXFTCLN CLEANUP DAEMON BPXFTCLN
BPXFTSYN SYNC DAEMON BPXFTSYN
BPXFPINT PIPE BPXFPINT
BPXFCSIN CHAR SPECIAL BPXFCSIN
NFS REMOTE FILE SYSTEM GFSCINIT
PFS NAME DESCRIPTION ENTRY STATUS FLAGS
TCP41 SOCKETS EZBPFINI ACT CD
TCP42 SOCKETS EZBPFINI ACT
TCP43 SOCKETS EZBPFINI INACT SD
TCP44 SOCKETS EZBPFINI INACT
PFS PARM INFORMATION
HFS SYNCDEFAULT(60) FIXED(50) VIRTUAL(100)
CURRENT VALUES: FIXED(55) VIRTUAL(100)
NFS biod(6)
```

- **DISPLAY OMVS,PID=processid** オペレーター・コマンドは、特定のプロセスのスレッド情報を表示します。

```
d omvs,pid=16777456
BPX0040I 15.30.01 DISPLAY OMVS 637
OMVS 000E ACTIVE OMVS=(76)
USER JOBNAME ASID PID PPID STATE START CT_SECS
STCRSE RSED8 007E 16777456 67109106 HF---- 20.00.56 113.914
LATCHWAITPID= 0 CMD=java -Ddaemon.log=/var/rdz/logs -
THREAD_ID TCB@ PRI_JOB USERNAME ACC_TIME SC STATE
0E08A00000000000 005E6DF0 OMVS .927 RCV FU
0E08F00000000001 005E6C58 .001 PTX JYNV
0E09300000000002 005E6AC0 7.368 PTX JYNV
0E0CB00000000008 005C2CF0 OMVS 1.872 SEL JFNV
0E192000000003CE 005A0B70 OMVS IBMUSER 14.088 POL JFNV
0E18D000000003CF 005A1938 IBMUSER .581 SND JYNV
```

ネットワークのモニター

ホストに接続している多数のクライアントをサポートしている場合は、Developer for System z だけでなく、ネットワーク・インフラストラクチャーでもワークロードを処理することが必要です。ネットワーク管理は、Developer for System z 資料が扱う範囲外の対象で、これに関しては幅広くかつ数多くこれまでに解説されています。したがって、以下の指針のみを示しておきます。

- **DISPLAY NET,CSM** オペレーター・コマンドでは、通信ストレージ・マネージャー (CSM) で管理されているストレージの使用状況をモニターすることができます。このコマンドを使用して、ECSA およびデータ・スペース・ストレージ・プールに CSM ストレージがどの程度使用されているかを確認できます。詳細については、「*Communications Server SNA オペレーション*」(SC88-8930) を参照してください。

z/OS UNIX ファイル・システムのモニター

Developer for System z は、z/OS UNIX ファイル・システムを使用して、ログや一時ファイルなどのさまざまなタイプのデータを保管します。z/OS UNIX の **df** コマンドを使用すると、基礎となる HFS または zFS データ・セットの次のエクステントを作成する前に、まだ使用可能なファイル記述子の数とフリー・スペースの残量を確認することができます。

```
$ df
Mounted on      Filesystem      Avail/Total      Files      Status
/tmp            (OMVS.TMP)      1393432/1396800  4294967248  Available
/u/ibmuser      (OMVS.U.IBMUSER) 1248/1728        4294967281  Available
/usr/lpp/rdz    (OMVS.FEK.HHOP760) 3062/43200       4294967147  Available
/var            (OMVS.VAR)      27264/31680      4294967054  Available
```

サンプル・セットアップ

次のサンプル・セットアップでは、以下の要件をサポートするために必要な構成を示します。

- 同時クライアント接続数は 500
- 同時 MVS ビルド数 (バッチ・ジョブ) は 300
- 同時 CARMA 接続数 (CRASTART 始動方式を使用) は 200
- 3 時間の非アクティブ・タイムアウト
- z/OS UNIX の使用は不許可
- SCLM Developer Toolkit と File Manager Integration は使用しない
- Java ヒープの平均使用量を 5 MB と予測
- ユーザーに固有の z/OS UNIX UID を付与

スレッド・プールの数

デフォルトでは、Developer for System z は単一のスレッド・プールに 60 ユーザーを追加しようとします。ただし上記の要件では、非アクティブ・タイムアウトをアクティブにするように指定しています。このために、接続されているクライアントごとに 1 つずつスレッドが追加されることが 236 ページの表 44 からわかります。このスレッドはタイマー・スレッドであるため、常にアクティブです。したがって $60 \times (16+1) = 1020$ となり、かつ `maximum.threads` がデフォルトで 1000 に設定されているので、RSE は単一のスレッド・プールに 60 ユーザーを配置できなくなります。

`maximum.threads` を増やすことはできますが、ユーザー当たりの Java ヒープを平均 5 MB にするという要件があるため、`maximum.clients` を 50 まで下げることになりました。これで Java ヒープ・サイズの最大値がデフォルトの 256 MB 以内に収まります ($5 \times 50 = 250$)。

スレッド・プール当たりのクライアント数が 50 であり、サポートの必要な接続数が 500 であることから、必要なスレッド・プール・アドレス・スペースの数は 10 であることがわかりました。

最小限度の特定

この章で説明した数式と、このセクションの冒頭で示した基準を使用して、対応が必要なリソース使用量を特定することができます。

- アドレス・スペースの数 - 最大

$$3 + A + N*(x + y + z) + (2 + N*0.01)$$

$$3 + 10 + 500*1 + 200*1 + 300*1 + (2 + 500*0.01) = 1020$$

- アドレス・スペースの数 - ユーザー単位

$$x + y + z$$

$$1 + 1 + 1 = 3$$

- プロセスの数 - 最大

$$7 + 2*A + N*(x + y + z) + (10 + N*0.05)$$

$$7 + 2*10 + 500*2 + 200*1 + 300*0 + (10 + 500*0.05) = 1562$$

- プロセスの数 - ユーザー単位

$$(x + y + z) + 5*s$$

$$(2 + 1 + 0) + 5*0 = 3$$

- スレッドの数 - RSE スレッド・プール

$$9 + N*(16 + x + y + z) + (20 + N*0.1)$$

$$9 + 60*(16 + 1 + 4 + 0) + (20 + 60*0.1) = 1295$$

- スレッドの数 - JES ジョブ・モニター

$$3 + N$$

$$3 + 500 = 503$$

- ユーザー ID の数

$$500 + 3 = 503$$

追加されている 3 つのユーザー ID は、Developer for System z 開始タスクのユーザー ID である STCJMON、STCLOCK、および STCRSE です。

限度の定義

リソース使用量の数値が判明したので、限度を指定するディレクティブを適切な値でカスタマイズできます。

- /etc/rdz/rsed.envvars

- Xmx256m

変更なし

- Dmaximum.clients=50

- Dmaximum.threads=1000

変更なし

- Dminimum.threadpool.process=10

この変更はオプションです。RSE は必要に応じて新しいスレッド・プールを開始します。

- DHIDE_ZOS_UNIX=true
- DDSTORE_IDLE_SHUTDOWN_TIMEOUT=10800000
- FEK.#CUST.PARMLIB(FEJJCNFG)
 - MAX_THREADS=503
- SYS1.PARMLIB(BPXPRMxx)
 - MAXPROCSYS(2500)

最小は 1562、Developer for System z 以外のタスク用にバッファを追加

- MAXPROCUSER(25)

変更なし、最小は 3

- MAXTHREADS(1500)

ユーザー ID STCRSE の OMVS セグメント内の THREADSMAX を使用して RSE の限度 (最小は 1295) が設定されている場合、必ず最小は 503 (JES ジョブ・モニター用)

- MAXTHREADTASKS(1500)

ユーザー ID STCRSE の OMVS セグメント内の THREADSMAX を使用して RSE の限度 (最小は 1295) が設定されている場合、必ず最小は 503 (JES ジョブ・モニター用)

- MAXUIDS(700)

最小は 503、Developer for System z 以外のタスク用にバッファを追加

- MAXASSIZE(209715200)

変更なし (システム・デフォルトは 200 MB)、ここではユーザー ID STCRSE の OMVS セグメント内で ASSIZEMAX を使用

- SYS1.PARMLIB(IEASYSxx)
 - MAXUSER=2000

最小は 1020、Developer for System z 以外のタスク用にバッファを追加

- ユーザー ID STCRSE の OMVS セグメント
 - ASSIZEMAX(2147483647)

2 GB

リソース使用量のモニター

257 ページの『限度の定義』の説明に従ってシステム限度を定義したら、Developer for System z によるリソース使用量のモニターを開始して、変数の調整が必要かどうかを確認できます。259 ページの図 58 は、495 人のユーザーがログオンした後のリソース使用量を示しています (この図の例はログオンだけを示しています。ユー

ザー・アクションは示されていません)。

```
BPXM023I (STCRSE)
ProcessId(16779764) Memory Usage(10%) Clients(50) Order(1)
ProcessId(67108892) Memory Usage(16%) Clients(50) Order(2)
ProcessId(67108908) Memory Usage(10%) Clients(50) Order(3)
ProcessId(67108898) Memory Usage(16%) Clients(50) Order(4)
ProcessId(67108916) Memory Usage(16%) Clients(50) Order(5)
ProcessId(67108897) Memory Usage(16%) Clients(50) Order(6)
ProcessId(67108921) Memory Usage(16%) Clients(50) Order(7)
ProcessId(83886146) Memory Usage(16%) Clients(50) Order(8)
ProcessId(67108920) Memory Usage(16%) Clients(50) Order(9)
ProcessId(3622    ) Memory Usage(8%) Clients(45) Order(10)
```

Jobname	Cpu time	Storage	EXCP
-----	-----	-----	-----
JMON	1.74	43.0M	2753
LOCKD	10.05	31.9M	24621
RSED	6.65	40.1M	41780
RSED1	8.17	187.0M	76566
RSED2	13.04	184.9M	78946
RSED3	17.77	181.1M	76347
RSED4	11.63	174.9M	74638
RSED5	15.27	172.9M	72883
RSED6	13.85	180.8M	75031
RSED7	9.79	174.3M	76636
RSED8	21.59	176.1M	70583
RSED8	18.88	184.7M	76953
RSED9	9.52	189.8M	80490

図 58. サンプル・セットアップのリソース使用量

第 14 章 パフォーマンスに関する考慮事項

z/OS は高度にカスタマイズ可能なオペレーティング・システムであり、システムの場合によっては小さな) 変更で全体のパフォーマンスに大きな影響を与えることができます。この章では、Developer for System z のパフォーマンスを向上させるために行うことができるいくつかの変更を中心に説明します。

システムのチューニングの詳細については、「MVS 初期設定およびチューニングガイド」(SA88-8563)、および「UNIX System Services 計画」(GA88-8639)を参照してください。

zFS ファイル・システムの使用

zFS (zSeries® ファイル・システム) および HFS (階層ファイル・システム) は、どちらも z/OS UNIX 環境で利用できる UNIX ファイル・システムです。ただし、zFS には、以下のようなフィーチャーと利点があります。

- 多数のカスタマー環境で頻繁にアクセスおよび更新される、サイズが 8 K に近いファイルにアクセスする際のパフォーマンスの向上。これより小さいファイルのアクセス・パフォーマンスは、HFS のそれと同等です。
- 同じデータ・セット内のファイル・システムの読み取り専用クローン作成。クローン作成されたファイル・システムを各ユーザーが使用できるようにして、ファイル・システムの読み取り専用ポイント・イン・タイム・コピーを提供することができます。これはオプションのフィーチャーで、非シスプレックス環境でのみ使用できます。
- zFS は、戦略的な z/OS UNIX ファイル・システムです。HFS の機能は、既に固定化されており、ファイル・システムの機能拡張は zFS だけのものです。

zFS の詳細については、「UNIX System Services 計画」(GA88-8639)を参照してください。

STEPLIB の使用の回避

親から子へ、または exec を越えて伝搬される STEPLIB を持つ z/OS UNIX プロセスは、それぞれが約 200 バイトの拡張共通ストレージ域 (ECSA) を消費します。STEPLIB 環境変数が定義されなかった場合、または STEPLIB=CURRENT として定義された場合、z/OS UNIX は現在アクティブなすべての TASKLIB、STEPLIB、および JOBLIB 割り振りを、fork()、spawn()、または exec() 関数の実行中に伝搬します。

Developer for System z では、rsed.envvars 構成ファイルの中で説明されているように、STEPLIB=NONE がデフォルトとして rsed.envvars 内にコーディングされています。上記の理由から、このディレクティブを変更せず、代わりに、ターゲット・データ・セットを LINKLIST または LPA (リンク・パック域) の中に置くことをお勧めします。

システム・ライブラリーへのアクセスの改善

特定のシステム・ライブラリーとロード・モジュールは、z/OS UNIX およびアプリケーション開発アクティビティーによって頻繁に使用されます。それらをリンク・パック域 (LPA) に追加するなどの方法でアクセスを改善すると、システムのパフォーマンスを向上させることができます。以下で説明する SYS1.PARMLIB メンバーの変更について詳しくは、「MVS 初期設定およびチューニング解説書」(SA88-8564) を参照してください。

言語環境プログラム (LE) ランタイム・ライブラリー

C プログラム (z/OS UNIX シェルを含む) は、実行されると、言語環境プログラム (LE) ランタイム・ライブラリーからのルーチンを頻繁に使用します。平均すると、LE 対応プログラムを実行する 1 つのアドレス・スペースごとに約 4 MB のランタイム・ライブラリーがメモリーにロードされ、すべてのフォークにコピーされます。

CEE.SCEELPA

CEE.SCEELPA データ・セットには、z/OS UNIX によって頻繁に使用される LE ランタイム・ルーチンのサブセットが入っています。最大のパフォーマンス向上を実現するために、このデータ・セットを SYS1.PARMLIB(LPALSTxx) に追加する必要があります。そうすることにより、モジュールはディスクから 1 回だけ読み取られ、共用ケーションに保管されます。

注: ロード・モジュールを動的 LPA (リンク・パック域) に追加したい場合は、次のステートメントを SYS1.PARMLIB(PROGxx) に追加してください。

```
LPA ADD MASK(*) DSN(CEE.SCEELPA)
```

また、データ・セットを SYS1.PARMLIB(LNKLSTxx) または SYS1.PARMLIB(PROGxx) に追加することにより、LE ランタイム・ライブラリーを CEE.SCEERUN および CEE.SCEERUN2 に配置することもお勧めします。これにより、z/OS UNIX STEPLIB のオーバーヘッドがなくなり、LLA および VLF、またはそれらと同様な製品による管理のための入出力を削減できます。

注: 同じ理由から、C/C++ DLL クラス・ライブラリー CBC.SCLBDLL も、LINKLIST に追加してください。

これらのライブラリーを LINKLIST 内に置かない場合は、rsed.envvars 構成ファイルの中の説明に従い、適切な STEPLIB ステートメントを rsed.envvars 内にセットアップする必要があります。この方式では、常に追加の仮想ストレージが使用されますが、LE ランタイム・ライブラリーを LLA またはそれに類似した製品に対して定義することにより、パフォーマンスを改善できます。これにより、モジュールをロードするために必要な入出力が削減されます。

アプリケーション開発

アプリケーション開発が主なアクティビティーであるシステムでは、以下の行を SYS1.PARMLIB(PROGxx) に追加して、リンケージ・エディターを動的 LPA の中に置くと、パフォーマンスが向上する場合もあります。


```
LPA ADD MODNAME(CEEINIT,CEEELIB,CEEV003,EDCV) DSNAME(CEE.SCEERUN)
LPA ADD MODNAME(IEFIB600,IEFXB603) DSNAME(SYS1.LINKLIB)
```

C/C++ 開発用に、CBC.SCCNCMP コンパイラー・データ・セットを
SYS1.PARMLIB(LPALSTxx) に追加することもできます。

上記のステートメントは、考えられる LPA 候補の例ですが、必要性はご使用のサイトによって異なる場合があります。他の LE ロード・モジュールを動的 LPA 内に配置する方法については、「*Language Environment カスタマイズ*」(SA88-8552)を参照してください。C/C++ コンパイラー・ロード・モジュールを動的 LPA の中に配置する方法については、「*UNIX System Services 計画*」(GA88-8639)を参照してください。

セキュリティ検査のパフォーマンスの向上

z/OS UNIX について行われるセキュリティ検査のパフォーマンスを向上させるには、セキュリティ・ソフトウェアの FACILITY クラスで BPX.SAFFASTPATH プロファイルを定義します。これにより、さまざまな操作について z/OS UNIX セキュリティ検査を行うときのオーバーヘッドが削減されます。それらの検査には、ファイル・アクセス検査、IPC アクセス検査、およびプロセス所有権検査が含まれます。このプロファイルの詳細については、「*UNIX System Services 計画*」(GA88-8639)を参照してください。

注: ユーザーは、BPX.SAFFASTPATH プロファイルに対するアクセス権限を持っている必要はありません。

ワークロード管理

それぞれのサイトには固有の必要性があり、それを満たすために、z/OS オペレーティング・システムをカスタマイズして、使用可能なリソースを最大限に活用することができます。ワークロード管理を使用すると、パフォーマンスの最終目標を定義し、各目標にビジネスの重要度を割り当てることができます。作業の最終目標をビジネス用語で定義し、システムでは、その目標を達成するために、CPU やストレージなどのリソースをどれくらい作業に割り当てればよいかを決定します。

Developer for System z のパフォーマンスは、プロセスに正しい目標を設定することによってバランスをとることができます。以下に、いくつかの一般ガイドラインを示します。

- APPC トランザクションを使用する場合は、それを TSO パフォーマンス・グループに割り当てます。
- 開始タスク・パフォーマンス・グループ (SYSSTC) を Developer for System z サーバー・アドレス・スペース (JES ジョブ・モニター (JMON)、ロック・デモン (LOCKD)、RSE デモン (RSED)、および RSE スレッド・プール (RSEDx)) に割り当てます。

この件について詳しくは、「*MVS 計画: ワークロード管理*」(SA88-8574)を参照してください。

固定 Java ヒープ・サイズ

固定サイズのヒープを使用すると、ヒープの拡張や縮小が発生しないため、状態によっては、パフォーマンスの大幅な向上につながります。ただし、固定サイズのヒープを使用することは、通常ではお勧めできません。その理由は、ヒープが満杯になるまでガーベッジ・コレクションの開始が遅延し、開始された時点では、それがメジャー・タスクになるからです。また、フラグメント化の危険も増し、ヒープの圧縮が必要になります。したがって、固定サイズのヒープは、適正なテストの後か、IBM サポートからの指示の下でのみ使用してください。ヒープ・サイズとガーベッジ・コレクションの詳細については、「*Java Diagnostics Guide*」(SC34-6650)を参照してください。

デフォルトでは、z/OS Java 仮想マシン (JVM) の初期ヒープ・サイズは、1 メガバイトです。最大サイズは 64 メガバイトです。限度は、-Xms (初期) および -Xmx (最大) Java コマンド行オプションで設定できます。

Developer for System z では、Java コマンド行オプションは、46 ページの『_RSE_JAVAOPTS での追加 Java 始動パラメーターの定義』の説明のように、rsed.envvars の _RSE_JAVAOPTS ディレクティブで定義されます。

```
#_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Xms128m -Xmx128m"
```

Java -Xquickstart オプション

注: Java -Xquickstart は、RSE サーバーに REXEC/SSH 代替始動方式を使用する場合にのみ役立ちます。この方式については、109 ページの『(オプション) REXEC (または SSH) の使用』に説明があります。

-Xquickstart オプションは、一部の Java アプリケーションの起動時間を改善するために使用できます。-Xquickstart を指定すると、JIT (Just In Time) コンパイラーが最適化のサブセット、つまりクイック・コンパイルを使用して実行されます。このクイック・コンパイルでは、起動時間を改善できます。

-Xquickstart は、実行期間が短いアプリケーション、特に実行時間が少数のメソッドに集中していないアプリケーションに適しています。-Xquickstart は、ホット・メソッドを含んでいる実行期間が長いアプリケーションに対して使用すると、パフォーマンスを低下させる場合があります。

RSE サーバー用に -Xquickstart オプションを有効にするには、次のディレクティブを rsed.envvars の末尾に追加します。

```
_RSE_JAVAOPTS="$_RSE_JAVAOPTS -Xquickstart"
```

JVM 間でのクラス共用

IBM Java 仮想マシン (JVM) バージョン 5 以上では、ブートストラップ・クラスおよびアプリケーション・クラスを共用メモリー内のキャッシュに保管することにより、それらを JVM 間で共用できます。クラス共用は、複数の JVM がキャッシュを共用している場合、全体的な仮想メモリーの消費を削減します。また、クラス共用はキャッシュが作成された後、JVM の起動時間を短縮します。

共用クラス・キャッシュはアクティブな JVM に依存せず、キャッシュを作成した JVM の存続期間を越えて持続します。共用クラス・キャッシュは JVM の存続期間を越えて持続するので、JAR またはファイル・システム上のクラスに加えられたすべての変更が反映されるよう、キャッシュは動的に更新されます。

新しいキャッシュの作成とデータの取り込みを行うためのオーバーヘッドは最小限で済みます。時間的な JVM 始動コストは、単一の JVM の場合、ロードされるクラスの数によって異なりますが、クラス共用を使用しないシステムに比べて 0 から 5 % の低下が一般的です。キャッシュにデータが取り込まれた場合の JVM 起動時間の改善は、オペレーティング・システムおよびロードされるクラス数によって異なりますが、クラス共用を使用しないシステムに比べて 10 % から 40 % の高速化となって現れます。複数の JVM を並行して実行すると、全体的な起動時間がさらに改善されます。

クラス共用の詳細については、「*Java SDK and Runtime Environment User Guide*」を参照してください。

クラス共用の有効化

RSE サーバー用にクラス共用を有効にするには、次のディレクティブを `rse.envvars` の末尾に追加します。最初のステートメントは、グループ・アクセス権を持つ RSE という名前のキャッシュを定義しており、これは、クラス共用が失敗した場合でも RSE サーバーを始動できるようにします。2 番目のステートメントはオプションであり、キャッシュ・サイズを 6 メガバイトに設定します (システム・デフォルトは 16 MB です)。3 番目のステートメントは、クラス共用パラメーターを Java 始動オプションに追加します。

```
_RSE_CLASS_OPTS=-Xshareclasses:name=RSE,groupAccess,nonFatal
# _RSE_CLASS_OPTS="$ _RSE_CLASS_OPTS -Xscmx6m
_RSE_JAVA_OPTS="$ _RSE_JAVA_OPTS $ _RSE_CLASS_OPTS"
```

注: 『キャッシュ・セキュリティ』 で述べたように、共用クラスを使用するすべてのユーザーは、同じ 1 次グループ ID (GID) を持っている必要があります。つまり、それらのユーザーにはセキュリティ・ソフトウェア内で同じデフォルト・グループが定義されている必要があるか、さまざまなデフォルト・グループが OMVS セグメント内に同じ GID を持っています。

キャッシュ・サイズ制限

理論上の最大共用キャッシュ・サイズは 2 GB です。指定できるキャッシュのサイズは、システムで利用できる物理メモリーとスワップ・スペースの容量によって制限されます。プロセスの仮想アドレス・スペースは共用クラス・キャッシュと Java ヒープの間で共用されるので、Java ヒープの最大サイズを大きくすると、作成できる共用クラス・キャッシュのサイズが小さくなります。

キャッシュ・セキュリティ

共用クラス・キャッシュへのアクセスは、オペレーティング・システムの許可および Java セキュリティ許可によって制限されます。

デフォルトでは、クラス・キャッシュはユーザー・レベル・セキュリティを使用して作成されるため、キャッシュを作成したユーザーだけがキャッシュにアクセスできます。z/OS UNIX では、`groupAccess` というオプションがあり、これはキャッ

シュを作成したユーザーの 1 次グループに属するすべてのユーザーにアクセス権を与えます。しかし、使用されたアクセス・レベルに関係なく、キャッシュを破棄できるのは、そのキャッシュを作成したユーザーかルート・ユーザー (UID 0) だけです。

Java SecurityManager を使用した追加セキュリティー・オプションの詳細については、「*Java SDK and Runtime Environment User Guide*」を参照してください。

SYS1.PARMLIB(BPXPRMxx)

いくつかの SYS1.PARMLIB(BPXPRMxx) の設定は、共用クラスのパフォーマンスに影響します。誤った設定を使用すると、共用クラスが機能しなくなる可能性があります。また、それらの設定は、パフォーマンスへの影響を持つ場合もあります。これらのパラメーターのパフォーマンスへの影響と使用方法について詳しくは、「*MVS 初期設定およびチューニング解説書*」(SA88-8564) および「*UNIX System Services 計画*」(GA88-8639) を参照してください。共用クラスの操作に最も大きな影響を与える BPXPRMxx パラメーターは、以下のとおりです。

- MAXSHAREPAGES、IPCSHMSPAGES、IPCSHMMPAGES、および IPCSHMNSEGS

これらの設定は、JVM で使用できる共用メモリー・ページの量に影響します。31 ビット z/OS UNIX システム・サービスの共用ページ・サイズは、4 KB に固定されています。共用クラスは、デフォルトでは 16 MB のキャッシュを作成しようとしています。したがって、IPCSHMMPAGES は 4096 より大きく設定してください。

-Xscmx を使用してキャッシュ・サイズを設定する場合、JVM は最も近いメガバイト値まで切り上げを行います。使用するシステム上で IPCSHMMPAGES を設定するときは、このことを考慮する必要があります。

- IPCSEMNIIDS および IPCSEMNSEMS

これらの設定は、UNIX プロセスで使用できるセマフォの量に影響します。共用クラスは、IPC セマフォを使用して JVM 間の通信を行います。

ディスク・スペース

共用クラス・キャッシュは、システム上に存在するキャッシュの識別情報を保管するために、ディスク・スペースを必要とします。この情報は /tmp/javasharedresources に保管されます。識別情報ディレクトリーが削除された場合、JVM はシステム上の共用クラスを識別できないため、キャッシュの再作成が必要になります。

キャッシュ管理ユーティリティー

Java -Xshareclasses 行コマンドは、いくつかのオプションをとることができ、それらのオプションの一部はキャッシュ管理ユーティリティーです。そのうちの 3 つを以下のサンプルに示します (\$ は z/OS UNIX プロンプトです)。サポートされるコマンド行オプションの詳細については、「*Java SDK and Runtime Environment User Guide*」を参照してください。

```
$ java -Xshareclasses:listAllCaches
Shared Cache      OS shmid          in use            Last detach time
RSE               401412            0                 Mon Jun 18 17:23:16 2007
```

Could not create the Java virtual machine.

```
$ java -Xshareclasses:name=RSE,printStats
```

Current statistics for cache "RSE":

```
base address      = 0x0F300058
end address       = 0x0F8FFFF8
allocation pointer = 0x0F4D2E28
```

```
cache size        = 6291368
free bytes        = 4355696
ROMClass bytes    = 1912272
Metadata bytes    = 23400
Metadata % used   = 1%
```

```
# ROMClasses      = 475
# Classpaths      = 4
# URLs            = 0
# Tokens          = 0
# Stale classes   = 0
% Stale classes   = 0%
```

Cache is 30% full

Could not create the Java virtual machine.

```
$ java -Xshareclasses:name=RSE,destroy
JVMSHRC010I Shared Cache "RSE" is destroyed
Could not create the Java virtual machine.
```

注:

- キャッシュ・ユーティリティは、指定されたキャッシュに対して必要な操作を、JVM を始動せずに実行するので、「Could not create the Java virtual machine」というメッセージは正常です。
- キャッシュを破棄できるのは、そのキャッシュを使用しているすべての JVM のシャットダウンが完了しており、しかもコマンドを発行するユーザーが十分な権限を持っている場合だけです。

第 15 章 CICS TS に関する考慮事項

従来、CICS に対してリソースを定義する役割は、CICS 管理者の専門的な役割でした。アプリケーション開発者が CICS リソースを定義できるようにすることには、以下のようなさまざまな理由から抵抗がありました。

- ほとんどの CICS リソース定義には多数のパラメーターがあり、それらのパラメーターは、その複雑さ、他のリソース定義との相互関係、および作業基準のために、正しく定義するには CICS 管理者の知識が必要になります。定義が正しくないと、予期しない結果が生じる場合があります、それによって CICS 領域全体が影響を受けるおそれがあります。
- ほとんどのお客様の作業現場では、複数のアプリケーション・グループおよび開発者によって共用される必要がある CICS の開発環境とテスト環境が提供されています。多数のお客様の作業現場には、これらの環境についてのサービス・レベル・アグリーメントがあります。それらのアグリーメントを順守するには、環境を厳格に管理する必要があります。

Developer for System z では、この問題に対処するため、CICS 管理者が Application Deployment Manager の一部である CICS リソース定義 (CRD) サーバーを使用して、CICS リソース定義のデフォルトおよび CICS リソース定義パラメーターの表示プロパティを制御できるようにしています。

例えば、CICS 管理者は、アプリケーション開発者が更新できない特定の CICS リソース定義パラメーターを提供できます。その他の CICS リソース定義パラメーターは更新可能で、デフォルトが提供される場合もされない場合もあります。あるいは、無用な複雑さを避けるために、CICS リソース定義パラメーターを非表示にすることもできます。

アプリケーション開発者は、CICS リソース定義に満足できたら、それらの定義を稼働中の CICS テスト環境に直ちにインストールするか、さらに CICS 管理者による編集と承認を受けるために、マニフェストにエクスポートすることができます。CICS 管理者は、管理ユーティリティ (バッチ・ユーティリティ) またはマニフェスト処理ツールを使用して、リソース定義の変更を実装できます。

注: マニフェスト処理ツールは、IBM CICS エクスプローラー用のプラグインです。

ホスト・システムに Application Deployment Manager をセットアップするために必要となるタスクの詳細については、79 ページの『第 4 章 (オプション) Application Deployment Manager』を参照してください。

Application Deployment Manager のカスタマイズでは、以下のサービスが Developer for System z に追加されます。

- (クライアント側) IBM CICS Explorer™は、CICS リソースを表示および管理するための Eclipse ベースのインフラストラクチャーを提供し、CICS ツール同士のさらに緊密な統合を可能にします。
- (クライアント側) CICS リソース定義 (CRD) エディター

- (ホスト側) CICS リソース定義 (CRD) サーバー (CICS アプリケーションとして稼働)

Application Deployment Manager CICS リソース定義 (CRD) サーバーは、CRD サーバー自体と、CRD リポジトリ、関連する CICS リソース定義、および Web サービス・インターフェースを使用する場合は、Web サービス・バインド・ファイルとサンプルのパイプライン・メッセージ・ハンドラーから構成されます。CRD サーバーは、Developer for System z 資料で CICS 主接続領域として参照されている Web Owning Region (WOR) 内で稼働する必要があります。

現行リリースの Developer for System z で使用可能な Application Deployment Manager のサービスの詳細については、Developer for System z インフォメーション・センター (<http://publib.boulder.ibm.com/infocenter/ratdevz/v7r6/index.jsp>) を参照してください。

RESTful と Web サービス

CICS Transaction Server バージョン 4.1 以上では、Representational State Transfer (RESTful) の原則に従って設計された HTTP インターフェースをサポートしています。現在この RESTful インターフェースは、戦略的な CICSTS インターフェースとしてクライアント・アプリケーションで使用されています。従来の Web サービス・インターフェースはすでに安定化しており、今後は RESTful インターフェースのみが機能拡張の対象となります。

Application Deployment Manager は、この指示書に従い、Developer for System z バージョン 7.6 以上で新たに導入されたすべてのサービスに RESTful CRD サーバーを必要とします。

必要であれば、1 つの CICS 領域で RESTful インターフェースと Web サービス・インターフェースを同時にアクティブにすることができます。この場合、その領域で 2 つの CRD サーバーがアクティブになります。両サーバーは、同じ CRD リポジトリを共有します。2 番目のインターフェースを領域に対して定義すると、CICS から定義の重複に関する警告が発行されるので注意してください。

主接続領域と非主接続領域

CICS テスト環境は、いくつかの Multi-Region Option (MRO) 接続領域から構成される場合があります。時の経過と共に、それらの領域を分類するために非公式の指定が使用されてきました。一般的な指定は、Terminal Owning Region (TOR)、Web Owning Region (WOR)、Application Owning Region (AOR)、および Data Owning Region (DOR) です。

Web Owning Region は、CICS Web サービス・サポートを実装するために使用され、Application Deployment Manager CICS リソース定義 (CRD) サーバーをこの領域内で実行する必要があります。この領域は、Application Deployment Manager に対しては CICS 主接続領域として認識されます。CRD クライアントは、CICS 主接続領域への Web サービス接続を実装します。

CICS 非主接続領域は、CRD サーバーのサービス対象となる、それ以外のすべての領域です。このサービスには、IBM CICS エクスプローラーを使用したリソースの表示と、CICS リソース定義エディターを使用したリソースの定義が含まれます。

CICSplex SM ビジネス・アプリケーション・サービス (BAS) を使用して CICS 主接続領域の CICS リソース定義を管理する場合は、BAS によって管理される、その他のすべての CICS 領域を CRD サーバーのサービス対象とすることができます。

BAS で管理されていない CICS 領域を CRD サーバーによってサービス可能にするには、追加の変更が必要です。

CICS リソース・インストール・ロギング

CRD サーバーが CICS リソースに対して行ったアクションは、CICS CSDL TD キューの中にログとして記録されます。一般に、このキューは、使用している CICS 領域の DD MSGUSR を指します。

CICSplex SM ビジネス・アプリケーション・サービス (BAS) を使用して CICS リソース定義を管理する場合、ログを作成するには、CICSplex SM EYUPARM ディレクティブ BASLOGMSG を (YES) に設定する必要があります。

Application Deployment Manager セキュリティー

CRD リポジトリ・セキュリティ

CRD サーバー・リポジトリ VSAM データ・セットは、すべてのデフォルト・リソース定義を保持しています。したがって、更新されないように保護する必要がありますが、開発者は、そこに保管された値の読み取りを許可される必要があります。CRD リポジトリを保護するためのサンプルの RACF コマンドについては、192 ページの『データ・セット・プロファイルの定義』を参照してください。

パイプライン・セキュリティ

CICS が Web サービス・インターフェースを通じて SOAP メッセージを受信した場合、そのメッセージはパイプラインによって処理されます。パイプラインは順番に実行される一連のメッセージ・ハンドラーです。CICS は、パイプライン構成ファイルを読み取って、パイプライン内でどのメッセージ・ハンドラーを起動すればよいかを判別します。メッセージ・ハンドラーは、その中で、Web サービスの要求および応答の特別な処理を実行できるプログラムです。

Application Deployment Manager は、メッセージ・ハンドラーと SOAP ヘッダー処理プログラムの起動を指定する、サンプルのパイプライン構成ファイルを提供します。

パイプライン・メッセージ・ハンドラー (ADNTMSGH) は、SOAP ヘッダー内のユーザー ID とパスワードを処理することにより、セキュリティのために使用されます。ADNTMSGH は、サンプルのパイプライン構成ファイルによって参照されるため、CICS RPL 連結の中に入れる必要があります。

トランザクション・セキュリティ

CPIH はデフォルトのトランザクション ID で、パイプラインによって起動されたアプリケーションは、この ID の下で実行されます。多くの場合、CPIH は最小レベルの許可用に設定されます。

Developer for System z は、CICS リソースの定義および照会時に、CRD サーバーが使用する複数のトランザクションを提供します。これらのトランザクション ID は、要求された操作に応じて CRD サーバーが設定します。トランザクション ID のカスタマイズの詳細については、79 ページの『第 4 章 (オプション) Application Deployment Manager』を参照してください。

トランザクション	説明
ADMS	マニフェスト処理ツールからの CICS リソース変更要求用。一般に、これは CICS 管理者が使用するためのものです。このトランザクションは高いレベルの許可を必要とします。
ADMI	CICS リソースを定義、インストール、またはアンインストールする要求用。このトランザクションは、サイトのポリシーにもよりますが、中程度の許可を必要とすると考えられます。
ADMR	CICS の環境情報またはリソース情報を取り出す、上記以外のすべての要求用。このトランザクションは、サイトのポリシーにもよりますが、最小レベルの許可を必要とすると考えられます。

CRD サーバー・トランザクションによって行われるリソース定義要求の一部、または全部を、セキュリティで保護してください。最低でも、更新コマンド (デフォルトの Web サービス・パラメーター、デフォルトの記述子パラメーター、およびファイル名からデータ・セット名へのバインディング) をセキュリティで保護し、CICS 管理者のみが、グローバル・リソースのデフォルトの設定に使用されるこれらのコマンドを発行できるようにしてください。

トランザクションが接続すると、CICS リソース・セキュリティ検査機能 (使用可能な場合) により、ユーザー ID にはそのトランザクション ID を実行する許可が与えられます。

リソース検査は、稼働中のトランザクションの RESSEC オプション、RESSEC システム初期化パラメーターによって制御され、CRD サーバーの場合は XPCT システム初期化パラメーターによっても制御されます。

リソース検査は、XPCT システム初期化パラメーターの値が NO 以外で、かつ TRANSACTION 定義の RESSEC オプションが YES であるかまたは、RESSEC システム初期化パラメーターが ALWAYS である場合にのみ実行されます。

以下の RACF コマンドは、CRD サーバー・トランザクションを保護する方法の例を示しています。CICS セキュリティの定義の詳細については、「*RACF Security Guide for CICSTS*」を参照してください。

- RALTER GCICSTRN SYSADM UACC(NONE) ADDMEM(ADMS)
- PERMIT SYSADM CLASS(GCICSTRN) ID(#cicsadmin)
- RALTER GCICSTRN DEVELOPER UACC(NONE) ADDMEM(ADMI)
- PERMIT DEVELOPER CLASS(GCICSTRN) ID(#cicsdeveloper)
- RALTER GCICSTRN ALLUSER UACC(READ) ADDMEM(ADMR)
- SETROPTS RACLIST(TCICSTRN) REFRESH

SSL 暗号化通信

Application Deployment Manager クライアントが Web サービス・インターフェースを使用して CRD サーバーを起動するときは、データ・ストリームの SSL 暗号化がサポートされます。この通信への SSL の使用は、CICSTS TCIPSERVICE 定義の SSL(YES) キーワードによって制御されます。詳しくは、「*RACF Security Guide for CICSTS*」を参照してください。

リソース・セキュリティ

CICSTS は、リソースを保護する機能と、リソースを操作するコマンドを提供しています。セキュリティを完全に構成していない状態でアクティブにすると、一部の Application Deployment Manager アクションが失敗する場合があります (例えば、新しいリソース・タイプを操作する権限を付与するアクションなど)。

Application Deployment Manager で機能が失敗した場合は、以下のようなメッセージがないか CICS ログを調べ、「*RACF Security Guide for CICSTS*」の説明に従って修正を行ってください。

```
DFHXS1111 %date %time %applid %tranid Security violation by user
%userid at netname %portname for resource %resource in class
%classname. SAF codes are (X'safresp',X'safreas'). ESM codes are
(X'esmresp',X'esmreas').
```

管理ユーティリティ

Developer for System z が提供する管理ユーティリティを使用して、CICS 管理者は CICS リソース定義のデフォルト値を指定できます。これらのデフォルトは、読み取り専用とするか、アプリケーション開発者による編集を可能にすることができます。

管理ユーティリティは、以下の機能を提供します。

- CICSplex 管理対象テスト環境の CICSplex 名
- CICSplex SM ステージング・グループ名
- マニフェスト・エクスポート規則の設定
- CICS リソース属性のデフォルトおよび表示許可
- VSAM データ・セット定義に使用される、CICS 論理から物理へのバインディング

管理ユーティリティは、データ・セット FEK.#CUST.JCL 内のサンプル・ジョブ ADNJSAPU によって呼び出されます。このユーティリティを使用するには、CRD リポジトリに対する UPDATE アクセス権が必要です。

ADNJSPAU は FEK.#CUST.JCL に置かれます。ただし、z/OS システム・プログラマーが、ジョブ FEK.SFEKSAMP(FEKSETUP)をカスタマイズして実行依頼したときに別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

注: ADNJSPAU ジョブを実行する前に、CICS で CRDリポジトリをクローズする必要があります。このリポジトリは、ジョブの完了後に再びオープンすることができます。例えば、CICS にサインオンした後、以下のコマンドを入力して、ファイルをそれぞれクローズおよびオープンすることができます。

- CEMT S FILE(ADNREPF0) CLOSED
- CEMT S FILE(ADNREPF0) OPEN

CICS テスト環境の CRD リポジトリを更新するために、入力制御ステートメントが使用されます。この環境には、以下の一般的な構文規則が適用されます。

- 1 桁目のアスタリスクは、コメント行を示しています。
- DEFINE コマンドは 1 桁目から始める必要があり、直後に 1 つのスペース、その直後に有効なキーワード (TRANSACTION など) を続ける必要があります。
- キーワードの値は、キーワードの直後に続ける必要があります。間にスペースを入れることは許されません。唯一の例外は、表示許可キーワードの UPDATE、PROTECT、および HIDDEN の場合で、これらは値を持ちません。
- キーワードの値は、小括弧で囲みます。
- キーワードとその値は、1 行に収める必要があります。

以下のサンプル定義は、「*CICS Resource Definition Guide for CICSTS*」で定義されている DFHCSDUP コマンドの構造に従っています。唯一の相違点は、以下の表示許可キーワードが挿入されていることで、これらのキーワードは、属性値を 3 つの許可セットにグループ化するために使用されます。

UPDATE	このキーワードの直後にある属性は、Developer for System z を使用するアプリケーション開発者によって更新可能になります。これは、省略された属性のデフォルトでもあります。
PROTECT	このキーワードの直後にある属性は、表示されますが、Developer for System z を使用するアプリケーション開発者による更新から保護されます。
HIDDEN	このキーワードの直後にある属性は表示されず、Developer for System z を使用するアプリケーション開発者による更新からも保護されます。

以下の ADNJSPAU コード・サンプルを参照してください。

```

//ADNJSPAU JOB <JOB PARAMETERS>
//*
//ADNSPAU EXEC PGM=ADNSPAU,REGION=1M
//STEPLIB DD DISP=SHR,DSN=FEK.SFEKLOAD
//ADMREP DD DISP=OLD,DSN=FEK.#CUST.ADNREP00
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
*
* CICSplex SM parameters
*
DEFINE CPSMNAME( )
*DEFINE STAGINGGROUPNAME(ADMSTAGE)
*
* Manifest export rule
*
DEFINE MANIFESTEXPORTRULE(installOnly)
*
* CICS resource definition defaults
* Omitted attributes default to UPDATE.
*
* DB2TRAN default attributes
*
DEFINE DB2TRAN()
    UPDATE DESCRIPTION()
    ENTRY()
    TRANSID()
*
* DOCTEMPLATE default attributes
*
DEFINE DOCTEMPLATE()
    UPDATE DESCRIPTION()
    TEMPLATENAME()
    FILE() TSQUEUE() TDQUEUE() PROGRAM() EXITPGM()
    DDNAME(DFHHTML) MEMBERNAME()
    HFSFILE()
    APPENDCRLF(YES) TYPE(EBCDIC)
*
* File default attributes
*
DEFINE FILE()
    UPDATE DESCRIPTION()
    RECORDSIZE() KEYLENGTH()
    RECORDFORMAT(V) ADD(NO)
    BROWSE(NO) DELETE(NO) READ(YES) UPDATE(NO)
    REMOTESYSTEM() REMOTENAME()
    PROTECT DSNNAME() RLSACCESS(NO) LSRPOOLID(1) STRINGS(1)
    STATUS(ENABLED) OPENTIME(FIRSTREF)
    DISPOSITION(SHARE) DATABUFFERS(2) INDEXBUFFERS(1)
    TABLE(NO) MAXNUMRECS(NOLIMIT)
    READINTEG(UNCOMMITTED) DSNSHARING(ALLREQS)
    UPDATEMODEL(LOCKING) LOAD(NO)
    JNLREAD(NONE) JOURNAL(NO)
    JNLSYNCREAD(NO) JNLUPDATE(NO)
    JNLADD(NONE) JNLSYNCSWRITE(YES)
    RECOVERY(NONE) FWDRECOVLOG(NO)
    BACKUPTYPE(STATIC)
    PASSWORD() NSRGROUP()
    CFDTPOOL() TABLENAME()

```

図 59. ADNJSPAU - CICS管理ユーティリティ (1/3)

```

*
* Mapset default attributes
*
DEFINE MAPSET()
    UPDATE  DESCRIPTION()
    PROTECT RESIDENT(NO) STATUS(ENABLED)
           USAGE(NORMAL) USELPACOPY(NO)
** Processtype default attributes
*
DEFINE PROCESSTYPE()
    UPDATE  DESCRIPTION()
           FILE(BTS)
    PROTECT STATUS(ENABLED)
           AUDITLOG() AUDITLEVEL(OFF)
*
* Program default attributes
*
DEFINE PROGRAM()
    UPDATE  DESCRIPTION()
           CEDF(YES) LANGUAGE(LE370)
           REMOTESYSTEM() REMOTENAME() TRANSID()
    PROTECT API(CICSAPI) CONCURRENCY(QUASIRENT)
           DATALOCATION(ANY) DYNAMIC(NO)
           EXECCKEY(USER) EXECUTIONSET(FULLAPI)
           RELOAD(NO) RESIDENT(NO)
           STATUS(ENABLED) USAGE(NORMAL) USELPACOPY(NO)
    HIDDEN JVM(NO) JVMCLASS() JVMPROFILE(DFHJVMPR)
*
* TDQueue default attributes
*
DEFINE TDQUEUE()
    UPDATE  DESCRIPTION()
           TYPE(INTRA)
* Extra partition parameters
    DDNAME() DSNAME()
    REMOTENAME() REMOTESYSTEM() REMOTELength(1)
    RECORDSIZE() BLOCKSIZE(0) RECORDFORMAT(UNDEFINED)
    BLOCKFORMAT() PRINTCONTROL() DISPOSITION(SHR)
* Intra partition parameters
    FACILITYID() TRANSID() TRIGERRLEVEL(1)
    USERID()
* Indirect parameters
    INDIRECTNAME()
    PROTECT WAIT(YES) WAITACTION(REJECT)
* Extra partition parameters
    DATABUFFERS(1)
    SYSOUTCLASS() ERROROPTION(IGNORE)
    OPENTIME(INITIAL) REWIND(LEAVE) TYPEFILE(INPUT)
* Intra partition parameters
    ATIFACILITY(TERMINAL) RECOVSTATUS(NO)

```

図 59. ADNJSAPU - CICSTS 管理ユーティリティー (2/3)


```

*
* Transaction default attributes
*
DEFINE TRANSACTION()
    UPDATE  DESCRIPTION()
            PROGRAM()
            TWASIZE(0)
            REMOTESYSTEM() REMOTENAME() LOCALQ(NO)
    PROTECT PARTITIONSET() PROFILE(DFHCICST)
            DYNAMIC(NO) ROUTABLE(NO)
            ISOLATE(YES) STATUS(ENABLED)
            RUNAWAY(SYSTEM) STORAGECLEAR(NO)
            SHUTDOWN(DISABLED)
            TASKDATAKEY(USER) TASKDATALOC(ANY)
            BREXIT() PRIORITY(1) TRANCLASS(DFHTCL00)
            DTIMOUT(NO) RESTART(NO) SPURGE(NO) TPURGE(NO)
            DUMP(YES) TRACE(YES) CONFDATA(NO)
            OTSTIMEOUT(NO) WAIT(YES) WAITTIME(00,00,00)
            ACTION(BACKOUT) INDOUBT(BACKOUT)
            RESSEC(NO) CMDSEC(NO)
            TRPROF()
            ALIAS() TASKREQ()
            XTRANID() TPNAME() XTPNAME()
|
| *
| * URDIMAP attributes
| *
| DEFINE URIMAP()
|     UPDATE  USAGE(CLIENT)
|             DESCRIPTION()
|             PATH(/required/path)
|             TCPIPSERVICE()
|             TRANSACTION()
|             PROGRAM()
|     PROTECT ANALYZER(NOANALYZER)
|             ATOMSERVICE()
|             CERTIFICATE()
|             CHARACTERSET()
|             CIPHERS()
|             CONVERTER()
|             HFSFILE()
|             HOST(host.mycompany.com)
|             HOSTCODEPAGE()
|             LOCATION()
|             MEDIATYPE()
|             PIPELINE()
|             PORT(NO)
|             REDIRECTTYPE(NONE)
|             SCHEME(HTTP)
|             STATUS(ENABLED)
|             TEMPLATENAME()
|             USERID()
|             WEBSERVICE()
|
| *
| * Optional file name to VSAM data set name binding
| *
| *DEFINE DSBINDING() DSNAME()
| /*

```

図 59. ADNJSAPU - CICSTS 管理ユーティリティー (3/3)

管理ユーティリティーのマイグレーションに関する注

Developer for System z バージョン 7.6.1 では、管理ユーティリティーに URIMAP サポートが追加されています。URIMAP サポートを使用できるようにするには、

CRD リポジトリ VSAM データ・セットを最大レコード・サイズ 3000 で割り振る必要があります。Developer for System z バージョン 7.6.1 までは、サンプルの CRD リポジトリ割り振りジョブは、最大レコード・サイズ 2000 を使用します。

以前の CRD リポジトリを使用している場合に URIMAP サポートを使用可能にするには、以下のステップを実行します。

1. 既存の CRD リポジトリ FEK.#CUST.ADNREPF0 のバックアップを作成します。
2. 既存の CRD リポジトリを削除します。
3. 新しい CRD リポジトリの割り振りと初期化を行うために、ジョブ FEK.SFEKSAMP(ADNVCRD) をカスタマイズして実行依頼します。カスタマイズの手順については、メンバー内のドキュメンテーションを参照してください。
4. 管理ユーティリティを使用して新しい CRD リポジトリにデータを取り込むために、ジョブ FEK.SFEKSAMP(ADNJSPAU) をカスタマイズして実行依頼します。

注:

- 既存の CRD リポジトリをマイグレーションする必要はありません。これは、管理ユーティリティが、実行されるたびに CRD リポジトリの内容全体を置換するからです。
- CRD リポジトリにはバージョンの互換性の問題がありません。サポートされる Developer for System z クライアント・コードおよびホスト・コードはすべて、どちらの最大レコード・サイズにも対応します。ただし、最大レコード・サイズが 3000 ではない場合は、URIMAP サポートが使用不可になります。

管理ユーティリティのメッセージ

以下のメッセージは、管理ユーティリティが SYSPRINT DD に対して発行します。メッセージ CRAZ1803E、CRAZ1891E、CRAZ1892E、および CRAZ1893E は、ファイル状況コード、VSAM 戻りコード、VSAM 機能コード、および VSAM フィードバック・コードを含んでいます。VSAM 戻りコード、機能コード、およびフィードバック・コードについては、「*DFSMS Macro Instructions for Data Sets*」(SC26-7408) に説明があります。ファイル状況コードについては、「*Enterprise COBOL for z/OS 言語解説書*」(SC88-9117) に説明があります。

CRAZ1800I

行 <最後の制御ステートメントの行番号> で正常に完了しました。

(completed successfully on line <last control statement line number>)

説明: システム・プログラマー管理ユーティリティは、正常に完了しました。

ユーザー応答: なし。

CRAZ1801W

行 <最後の制御ステートメントの行番号> で警告によって完了しました。

(completed with warnings on line <last control statement line number>)

説明: システム・プログラマー管理ユーティリティは、制御ステートメントの処理時に 1 つ以上の警告を検出して完了しました。

ユーザー応答: 他の警告メッセージを調べてください。

CRAZ1802E

行 <行番号> でエラーを検出しました。(encountered an error on line <line number>)

説明: システム・プログラマー管理ユーティリティは、重大エラーを検出しました。

ユーザー応答: 他の警告メッセージを調べてください。

CRAZ1803E

リポジトリ・オープン・エラー、状況=<ファイル状況コード>
RC=<VSAM 戻りコード> FC=<VSAM 機能コード> FB=<VSAM フィード
バック・コード> (Repository open error, status=<file status code>
RC=<VSAM return code> FC=<VSAM function code> FB=<VSAM
feedback code>)

説明: システム・プログラマー管理ユーティリティは、CRD リポジトリ
をオープンしようとして重大エラーを検出しました。

ユーザー応答: VSAM 状況コード、戻りコード、機能コード、およびフィ
ードバック・コードを調べてください。

CRAZ1804E

行 <行番号> の入力レコードを認識できませんでした。(Unrecognized
input record on line <line number>)

説明: システム・プログラマー管理ユーティリティは、認識できない入力
制御ステートメントを検出しました。

ユーザー応答: DEFINE コマンドの直後に 1 つのスペースと、その直後に
CPSMNAME、STAGINGGROUPNAME、MANIFESTEXPORTRULE、DSBINDING、DB2TRAN、
DOCTEMPLATE、FILE、MAPSET、PROCESSTYPE、PROGRAM、TDQUEUE、
TRANSACTION のいずれかのキーワードがあったかどうかを調べてください。

CRAZ1805E

行 <行番号> のキーワード <キーワード> を処理しています。(Processing
keyword <keyword> on line <line number>)

説明: システム・プログラマー管理ユーティリティは、DEFINE キーワ
ード入力制御ステートメントを処理しています。

ユーザー応答: なし。

CRAZ1806E

行 <行番号> のマニフェスト・エクスポート規則が無効です。(Invalid
manifest export rule on line <line number>)

説明: システム・プログラマー管理ユーティリティは、無効なマニフェス
ト・エクスポート規則を検出しました。

ユーザー応答: MANIFESTEXPORTRULE キーワードの値が
「installOnly」、「exportOnly」、「both」のいずれかであることを確認して
ください。

CRAZ1807E

行 <行番号> に DSNAME キーワードがありません。(Missing DSNAME
keyword on line <line number>)

説明: システム・プログラマー管理ユーティリティーが処理しようとした DEFINE DSBINDING 制御ステートメントに DSNAME キーワードがありませんでした。

ユーザー応答: DEFINE DSBINDING 制御ステートメントに DSNAME キーワードが含まれているかどうか調べてください。

CRAZ1808E

行 <行番号> のキーワード <キーワード> のキーワード値が無効です。
(Invalid keyword value for keyword <keyword> on line <line number>)

説明: システム・プログラマー管理ユーティリティーが DEFINE 制御ステートメントを処理しようとして、指定されたキーワードに無効な値を検出しました。

ユーザー応答: 指定されたキーワードの長さや値が正しいかどうか調べてください。

CRAZ1890W

行 <行番号> でのキーワード構文エラー。 (Keyword syntax error on line <line number>)

説明: システム・プログラマー管理ユーティリティーが DEFINE 制御ステートメントを処理しようとして、キーワードまたはキーワード値の構文エラーを検出しました。

ユーザー応答: キーワード値が小括弧で囲まれていて、キーワードの直後に存在することを確認してください。キーワードおよびキーワード値は両方とも同じ行になければなりません。

CRAZ1891W

リポジトリ重複キー書き込みエラー、状況=<ファイル状況コード>
RC=<VSAM 戻りコード> FC=<VSAM 機能コード> FB=<VSAM フィードバック・コード>。
(Repository duplicate key write error, status=<file status code> RC=<VSAM return code> FC=<VSAM function code> FB=<VSAM feedback code>)

説明: システム・プログラマー管理ユーティリティーは、CRD リポジトリに書き込もうとして重複キー・エラーを検出しました。

ユーザー応答: VSAM 状況コード、戻りコード、機能コード、およびフィードバック・コードを調べてください。

CRAZ1892W

リポジトリ書き込みエラー、状況=<ファイル状況コード> RC=<VSAM 戻りコード> FC=<VSAM 機能コード> FB=<VSAM フィードバック・コード>。
(Repository write error, status=<file status code> RC=<VSAM return code> FC=<VSAM function code> FB=<VSAM feedback code>)

説明: システム・プログラマー管理ユーティリティーは、CRD リポジトリに書き込もうとして重大エラーを検出しました。

ユーザー応答: VSAM 状況コード、戻りコード、機能コード、およびフィードバック・コードを調べてください。

CRAZ1893W

リポジトリ読み取りエラー、状況=<ファイル状況コード> RC=<VSAM

戻りコード> FC=<VSAM 機能コード> FB=<VSAM フィードバック・コード>。(Repository read error, status=<file status code> RC=<VSAM return code> FC=<VSAM function code> FB=<VSAM feedback code>)

説明: システム・プログラマー管理ユーティリティーは、CRD リポジトリから読み取ろうとして重大エラーを検出しました。

ユーザー応答: VSAM 状況コード、戻りコード、機能コード、およびフィードバック・コードを調べてください。

第 16 章 TSO 環境のカスタマイズ

この付録は、Developer for System z で TSO 環境に DD ステートメントとデータ・セットを追加することにより、TSO ログオン・プロシーチャーを模倣するのに役立ちます。

TSO コマンド・サービス

TSO コマンド・サービスは、TSO コマンドと (バッチ) ISPF コマンドを実行し、結果を要求側クライアントへ返す Developer for System z コンポーネントです。これらのコマンドは、本製品が非明示的に要求するか、ユーザーが明示的に要求できます。

Developer for System z で提供されるサンプル・メンバーは、最小の TSO/ISPF 環境を作成します。ご使用のワークショップ内の開発者がカスタム・ライブラリーまたはサード・パーティー・ライブラリーへのアクセスを必要とする場合、z/OS システム・プログラマーは必要な DD ステートメントおよびライブラリーを TSO コマンド・サービス環境に追加する必要があります。Developer for System z での実装は異なりますが、その背後にあるロジックは TSO ログオン・プロシーチャーと同一です。

注: TSO コマンド・サービスは非対話式のコマンド行ツールです。このため、データの入力を要求したり ISPF パネルを表示したりするコマンドまたはプロシーチャーは、機能しません。これらを実行するためには、3270 エミュレーター (Developer for System z クライアントの一部である Host Connect Emulator など) が必要です。

アクセス方式

バージョン 7.1 以降の Developer for System z では、TSO コマンド・サービスへのアクセス方法を選択できます。

- ISPF の TSO/ISPF クライアント・ゲートウェイ・サービス。これは最小の ISPF サービス・レベルを必要とします。これが、提供されたサンプルで使用されるデフォルトの方式です。
- APPC トランザクション (バージョン 7.1 より前のリリースと同様)。

注: ISPF の TSO/ISPF クライアント・ゲートウェイ・サービスは、バージョン 7.1 で使用されていた SCLM Developer Toolkit 機能の代わりになります。

バージョン 7.1 以上のホストに、どのアクセス方式が使用されているかを判別するには、rsed.envvars を調べます。構成プロセスのときにデフォルトが使用された場合、rsed.envvars は /etc/rdz/ にあります。

- `_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DTSO_SERVER=APPC"` ステートメントが存在しない (または、デフォルトでコメント化されたままである) 場合は、ISPF の TSO/ISPF クライアント・ゲートウェイ・サービスが使用されます。

- `_RSE_JAVAOPTS="$_RSE_JAVAOPTS -DTSO_SERVER=APPC"` ステートメントが存在する (しかも、コメント化されていない) 場合は、APPC が使用されます。

TSO/ISPF クライアント・ゲートウェイ・アクセス方式の使用

基本カスタマイズ - ISPF.conf

ISPF.conf 構成ファイル (デフォルトでは `/etc/rdz/` に置かれます) は、Developer for System z に使用する TSO/ISPF 環境を定義します。アクティブな ISPF.conf 構成ファイルは 1 つだけ存在し、すべての Developer for System z ユーザーによって使用されます。

この構成ファイルのメイン・セクションでは、次のサンプルに示すように、DD 名および関連するデータ・セット連結が定義されています。

```
sysproc=ISP.SISPLIB,FEK.SFEKPROC
ispmlib=ISP.SISPMENU
isptlib=ISP.SISPTENU
ispplib=ISP.SISPPENU
ispslib=ISP.SISPSLIB
ispllib=ISP.SISPLOAD
myDD=HLQ1.LLQ1,HLQ2.LLQ2
```

- 1 つの DD 定義は正確に 1 行を使用し (複数行はサポートされていません)、行の長さに制限はありません。
- 定義に大/小文字の区別はなく、空白文字はすべて無視されます。
- コメント行はアスタリスク (*) で始まります。
- DD 名の直後には等号 (=) があり、その直後にデータ・セット連結が続きます。複数のデータ・セット名は、コンマ (,) によって分離されます。
- データ・セット連結は、リストされた順序で検索されます。
- データ・セットは完全修飾でなければならない、引用符 (') で囲まれていたり変数を使用していたりしてはなりません。
- すべてのデータ・セットは、DISP=SHR によって割り振られます。
- 新しい DD 名は、任意に追加できますが、DD 名の (JCL) 規則に従っている必要があり、ISPF.conf 内にある別の構成パラメーターと競合してはなりません。また、TSO/ISPF クライアント・ゲートウェイ・サービスによって、ISPPROF が動的に割り振られます (DISP=NEW,DELETE)。

詳細設定 - 既存の ISPF プロファイルの使用

デフォルトでは、TSO/ISPF クライアント・ゲートウェイは TSO コマンド・サービス用に一時的な ISPF プロファイルを作成します。しかし、既存の ISPF プロファイルのコピーを使用するように、TSO/ISPF クライアント・ゲートウェイ z に指示することができます。ここで重要なものは、`rsed.envvars` 内の `_RSE_CMDSERV_OPTS` ステートメントです。

```
#_RSE_CMDSERV_OPTS="$_RSE_CMDSERV_OPTS &ISPPROF=&SYSUID..ISPPROF"
```

この機能を使用するには、このステートメントをコメント解除 (先行ポンド記号 (#) 文字を除去) し、既存の ISPF プロファイルの完全修飾データ・セット名を指定します。

データ・セット名の中で、以下の変数を使用できます。

- &SYSUID。開発者のユーザー ID の代わりに使用します。
- &SYSREF。開発者の TSO 接頭部の代わりに使用します。

注:

- "ISPPROF" で渡されたデータ・セット名が無効な場合は、一時的な空の ISPF プロファイルが代わりに使用されます。
- ISPF プロファイル (一時プロファイルとコピーされたプロファイルの両方) は、セッションの終了時に削除されます。プロファイルに加えられた変更は、既存の ISPF プロファイルにマージされません。

詳細設定 - 割り振り exec の使用

ISPF.conf 内の allocjob ステートメント (これは、デフォルトではコメント化されています) は、ある exec を指しており、この exec を使用すると、ユーザー ID 別に詳細なデータ・セット割り振りを指定できます。

```
*allocjob = FEK.#CUST.CNTL(CRAISPRX)
```

この機能を使用するには、このステートメントをコメント解除 (先行アスタリスク (*) 文字を除去) し、割り振り exec への完全修飾参照を指定してください。

- この exec は ISPPROF、および ISPF.conf 内で定義された DD の割り振りの後で、ただし ISPF が初期化される前に、実行されます。割り振り exec によってこれらの定義が取り消されないようにしてください。
- この exec には 1 つのパラメーター、つまり呼び出し元のユーザー ID が渡されます。
- サンプル exec の CRAISPRX がサンプル・ライブラリー FEK.#CUST.CNTL に入っています。ただし、ジョブ FEK.SFEKSAMP(FEKSETUP) をカスタマイズして実行依頼したときに、別のロケーションを指定した場合は除きます。詳細については、18 ページの『カスタマイズのセットアップ』を参照してください。

注: この exec は ISPF が初期化される前に呼び出されるので、**VPUT** および **VGET** は使用できません。しかし、PDS(E) または VSAM ファイルを使用して、これらの機能の独自の実装を作成できます。

詳細設定 - 複数の割り振り exec の使用

ISPF.conf は 1 つの割り振り exec の呼び出しだけをサポートしていますが、その exec が別の exec を呼び出すことに制限はありません。また、パラメーターとして渡されるクライアントのユーザー ID によって、個人別の割り振り exec を呼び出すことができます。例えば、メンバー USERID'.EXEC(ALLOC)'. が存在するかどうかを検査し、実行することができます。

この機能を上手に利用すれば、次のようにして既存の TSO ログオン・プロシージャーを使用できます。

- ユーザー固有の構成ファイル (USERID'.FEKPROF' など) を読み取ります。
- どのログオン・プロシージャーがファイル内に記述されているかを調べます。
- 記述されているプロシージャーを SYS1.PROCLIB から読み取り、構文解析して、中の DD ステートメントとデータ・セット割り振りを見つけます。

- データ・セットを実際のログオン・プロシージャーと同様な方法で割り振ります。

詳細設定 - 複数の Developer for System z セットアップでの複数の ISPF.conf ファイル

上記の割り振り exec シナリオで特定の要求を処理できない場合は、Developer for System z の RSE 通信サーバーのさまざまなインスタンスを作成し、それぞれに独自の ISPF.conf ファイルを使用することができます。以下に述べる方式の主な欠点は、Developer for System z ユーザーが、希望の TSO 環境を得るために同じホスト上の異なるサーバーに接続する必要があることです。

注: RSE サーバーの 2 番目のインスタンスを作成するために必要なことは、各構成ファイル、始動 JCL、および開始タスク定義を複製して更新することだけです。製品の新規インストールは必要なく、コードを複製する必要もありません。

```
$ cd /etc/rdz
$ mkdir /etc/rdz/tso2
$ cp rsed.envvars /etc/rdz/tso2
$ cp ISPF.conf /etc/rdz/tso2
$ ls /etc/rdz/tso2
ISPF.conf      rsed.envvars
$ oedit /etc/rdz/tso2/rsed.envvars
-> change: _CMDSERV_CONF_HOME=/etc/rdz/tso2
-> uncomment and change: -Ddaemon.log=/var/rdz/logs/tso2
-> add at the END:
# -- NEEDED TO FIND THE REMAINING CONFIGURATION FILES
CFG_BASE=/etc/rdz
CLASSPATH=.:$CFG_BASE:$CLASSPATH
# --
$ oedit /etc/rdz/tso2/ISPF.conf
-> change: change as needed
```

前の例のコマンドは、変更が必要な Developer for System z 構成ファイルを、新規に作成された tso2 ディレクトリーにコピーします。rsed.envvars 内の _CMDSERV_CONF_HOME 変数を更新して新しい ISPF.conf ホーム・ディレクトリーを定義し、daemon.log を更新して新しいログ・ロケーション（これは、存在しなければ自動的に作成されます）を定義する必要があります。CLASSPATH の更新は、tso2 にコピーされなかった構成ファイルを RSE が確実に検出できるようにします。ISPF.conf ファイル自体を、必要性に合わせて更新できます。ISPF 作業域 (rsed.envvars 内の変数 _CMDSERV_WORK_HOME) を両方のインスタンスで共用できることに注意してください。

この時点で残っている作業は、新しいポート番号と新しい /etc/rdz/tso2 構成ファイルを使用する RSE 用の新規開始タスクを作成することです。

上記の各アクションの詳細については、この資料の関連するセクションを参照してください。

APPC アクセス方式の使用

基本カスタマイズ - APPC トランザクション JCL

APPC トランザクションの定義は、APPC パラメーターとトランザクション JCL からなっています。Developer for System z APPC トランザクションを作成するサンプル JCL、FEK.#CUST.JCL(FEKAPPC) は、ISPF サポート付きと ISPF サポートなしでトランザクション JCL を定義する 2 つのオプションを保持しています。

```
//SYSIN DD DDNAME=SYSINISP * use SYSINTSO or SYSINISP
```

クライアントはトランザクション JCL 内で定義された TSO/ISPF 環境を取得するので、このセクションを通常の DD 規則に従ってカスタマイズすることにより、クライアントの環境をカスタマイズできます。

```
...
//CMDSERV EXEC PGM=IKJEFT01,DYNAMNBR=50,
//      PARM='ISPSTART CMD(%FEKFRSRV TIMEOUT=60) NEWAPPL(ISR) NESTMACS'
//SYSPROC DD DISP=SHR,DSN=FEK.SFEKPROC
//ISPPLIB DD DISP=SHR,DSN=ISP.SISPPENU
//ISPLLIB DD DISP=SHR,DSN=ISP.SISPMENU
//ISPTLIB DD DISP=SHR,DSN=ISP.SISPTENU
//ISPSLIB DD DISP=SHR,DSN=ISP.SISPSENU
//ISPPROF DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
//      SPACE=(TRK,(1,1,5)),LRECL=80,RECFM=FB
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN DD DUMMY
//MYDD DD DISP=SHR,DSN=HLQ1.LLQ1
//      DISP=SHR,DSN=HLQ2.LLQ2
```

注: 既存の APPC トランザクションは、APPC ISPF パネルを使用して変更できません。

詳細設定 - 既存の ISPF プロファイルの使用

ISPF サポートを選択した場合、デフォルトでは、Developer for System z は TSO コマンド・サービス用に一時的な ISPF プロファイルを作成します。しかし、既存の ISPF プロファイルのコピーを使用するように、Developer for System z に指示することができます。FEK.SFEKSAMP(FEKAPPC) サンプル・ジョブ内で説明されているように、以下のことを行う必要があります。

- トランザクション JCL 内で COPY ステップ (EXEC および関連する DD カード) をコメント解除します。
- &SYSUID..ISPPROF をユーザーの ISPF プロファイル・データ・セット名に合わせて変更します。
- CMDSERV ステップ内の最初の ISPPROF DD ステートメントをコメント化し、2 番目の DD ステートメントをコメント解除します。

```
...
//COPY EXEC PGM=IEBCOPY * (optional) clone existing ISPF profile
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DISP=SHR,DSN=&SYSUID..ISPPROF
//SYSUT2 DD DISP=(MOD,PASS),DSN=&&PROF,
//      UNIT=SYSALLDA,
//      LIKE=&SYSUID..ISPPROF
//*
//CMDSERV EXEC PGM=IKJEFT01,DYNAMNBR=50,
//      PARM='ISPSTART CMD(%FEKFRSRV TIMEOUT=60) NEWAPPL(ISR) NESTMACS'
```

```

// *ISPPROF DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
// *          SPACE=(TRK,(1,1,5)),LRECL=80,RECFM=FB
//ISPPROF DD DISP=(OLD,DELETE,DELETE),DSN=&&PROF

```

注: 無効なデータ・セット名が使用されている場合、APPC トランザクション (および、TSO コマンド・サービス) の始動は失敗します。

詳細設定 - 割り振り exec の使用

サンプルのトランザクション JCL は TSO コマンド・サービスを直接呼び出し、その際、名前 (FEKFRSRV) をパラメーターとしてプログラム IKJEFT01 に渡します。これを変更して、別の exec を呼び出すことができます。その exec は現行ユーザー ID に基づいて割り振りを行い、その後、TSO コマンド・サービスを呼び出すことができます。

TSO/ISPF クライアント・ゲートウェイ・アクセス方式とは逆に、この exec は環境のカスタマイズを支援するために、ユーザーの ISPF プロファイルに保管された変数を使用できます。ただし、プロファイルに対する更新は、実際のプロファイルでなく一時的なコピーを使用しているので、セッション終了時に失われることに留意してください。

ただし、APPC トランザクション内での割り振り exec の使用はサポートされておらず、上記の説明には保証がないことに注意してください。

詳細設定 - 複数の Developer for System z セットアップでの複数の APPC トランザクション

固有な複数の TSO 環境を必要とする場合は、Developer for System z の RSE 通信サーバーのさまざまなインスタンスを作成し、それぞれに独自の APPC トランザクションを使用することができます。以下に述べる方式の主な欠点は、Developer for System z ユーザーが、希望の TSO 環境を得るために同じホスト上の異なるサーバーに接続する必要があることです。

注: RSE サーバーの 2 番目のインスタンスを作成するために必要なことは、各構成ファイル、始動 JCL、および開始タスク定義を複製して更新することだけです。製品の新規インストールは必要なく、コードを複製する必要もありません。

```

$ cd /etc/rdz
$ mkdir /etc/rdz/tso2
$ cp rsed.envvars /etc/rdz/tso2/
$ ls /etc/rdz/tso2/
rsed.envvars
$ oedit /etc/rdz/tso2/rsed.envvars
-> uncomment and change: _FEKFSCMD_TP_NAME=_FEKFTS02
-> uncomment and change: -Ddaemon.Log=/var/rdz/logs/tso2
-> add at the END:
# -- NEEDED TO FIND THE REMAINING CONFIGURATION FILES
CFG_BASE=/etc/rdz
CLASSPATH=.:$CFG_BASE:$CLASSPATH
# --

```

上記のコマンドは、新しい tso2 ディレクトリーを作成し、変更が必要な Developer for System z 構成ファイルを新しいロケーションにコピーします。rsed.envvars 内の _FEKFSCMD_TP_NAME_ 変数を更新して、新しい APPC トランザクション名を定

義し、daemon.log を更新して新しいデーモン・ログ・ロケーション（これは、存在しなければ自動的に作成されます）を定義する必要があります。CLASSPATH の更新は、tso2 にコピーされなかった構成ファイルを RSE が確実に検出できるようにします。

```
//FEKAPPCC JOB CLASS=A,MSGCLASS=A,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//*
//TPADD EXEC PGM=ATBDSDFMU
//SYSSDLIB DD DISP=SHR,DSN=SYS1.APPCTP
//SYSSDOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN DD DDNAME=SYSINISP * use SYSINTSO or SYSINISP
//SYSINISP DD DATA,DLM='QT'
TPADD
TPNAME(FEKFTS02)
ACTIVE(YES)
TPSCHED_DELIMITER(DLM1)
KEEP_MESSAGE_LOG(ERROR)
MESSAGE_DATA_SET(&SYSUID..FEKFTS02.&TPDATE..&TPTIME..LOG)
DATASET_STATUS(MOD)
CLASS(A)
JCL_DELIMITER(DLM2)
//FEKFTS02 JOB CLASS=A,MSGCLASS=A,MSGLEVEL=(1,1)
//*
//CMDSERV EXEC PGM=IKJEFT01,DYNAMNBR=50,
// PARM='ISPSTART CMD(%FEKFRSRV TIMEOUT=60) NEWAPPL(ISR) NESTMACS'
//SYSPROC DD DISP=SHR,DSN=FEK.SFEKPROC
//ISPPLIB DD DISP=SHR,DSN=ISP.SISPPENU
//ISPLIB DD DISP=SHR,DSN=ISP.SISPMENU
//ISPTLIB DD DISP=SHR,DSN=ISP.SISPTENU
//ISPSLIB DD DISP=SHR,DSN=ISP.SISPSENU
//ISPPROF DD DISP=(NEW,DELETE,DELETE),UNIT=SYSALLDA,
// SPACE=(TRK,(1,1,5)),LRECL=80,RECFM=FB
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSTSIN DD DUMMY
//MYDD DD DISP=SHR,DSN=HLQ1.LLQ1
// DISP=SHR,DSN=HLQ2.LLQ2
DLM2
DLM1
QT
```

図 60. FEKAPPCC - 2 番目の APPC トランザクション

次に、上記のサンプルに示すように、サンプル・ジョブ FEK.#CUST.JCL(FEKAPPCC) をカスタマイズして実行依頼することにより、新しい APPC トランザクションを作成します。通常の (JCL 内で記述されている) カスタマイズのほかに、TPNAME を TPNAME(FEKFTS02) に変更して、新しい rsed.envvars 内の _FEKFSCMD_TP_NAME_ 定義に合わせる必要もあります。また、MESSAGE_DATA_SET 変数内の名前と、トランザクション JCL の JOB 名を変更する必要があります。

この時点で残っている作業は、新しいポート番号と新しい /etc/rdz/tso2 構成ファイルを使用する RSE 用の新規開始タスクを作成することです。

上記の各アクションの詳細については、この資料の関連するセクションを参照してください。

第 17 章 複数のインスタンスの実行

同じシステム上で Developer for System z の複数のインスタンスをアクティブにしたい場合があります。例えば、アップグレードをテストするときなどです。しかし、TCP/IP ポートなど、一部のリソースは共用できないため、デフォルトが常に適用可能であるとは限りません。この付録の情報を使用して Developer for System z の異なるインスタンスの共存を計画してください。その後、この構成ガイドを使用して、それらのインスタンスをカスタマイズできます。

Developer for System z の特定の部分は、2 つ (以上) のインスタンスで共用が可能です。それらのソフトウェア・レベルが同一で、しかも変更点が構成メンバー内に限られている場合を除いて、共用はお勧めできません。Developer for System z には、オーバーラップしない複数のインスタンスを作成するためのカスタマイズの余地が十分に残されており、それらのフィーチャーを使用することを強くお勧めします。

注:

- FEK および /usr/lpp/rdz は、製品のインストール時に使用された高位修飾子およびパスです。FEK.#CUST、/etc/rdz、および /var/rdz は、製品のカスタマイズ時に使用されたデフォルトのロケーションです (詳細については、18 ページの『カスタマイズのセットアップ』を参照してください)。
- 製品の z/OS UNIX の部分をデプロイしやすくするために、Developer for System z を専用ファイル・システム (HFS または zFS) にインストールしてください。
- 専用ファイル・システムを使用できない場合は、z/OS UNIX の tar コマンドなどのアーカイブ・ツールを使用して z/OS UNIX ディレクトリーをシステム間で転送してください。これにより、Developer for System z のファイルとディレクトリーの属性 (プログラム制御など) が保存されます。

Developer for System z インストール・ディレクトリーをアーカイブおよび復元するための以下のサンプル・コマンドについて詳しくは、「UNIX System Services コマンド解説書」(SA88-8641) を参照してください。

- アーカイブ: `cd /SYS1/usr/lpp/rdz; tar -cSf /u/userid/rdz.tar`
- 復元: `cd /SYS2/usr/lpp/rdz; tar -xSf /u/userid/rdz.tar`

シスプレックス全体での同一セットアップ

Developer for System z の構成ファイル (およびコード) は、シスプレックス内の複数のシステム間で共用でき、いくつかのガイドラインに従っていれば、各システムが Developer for System z の同一コピーをそれぞれに保持して実行することができます。

- ログ・ファイルを固有のロケーションに配置して、あるシステムが別のシステムの情報を上書きしないようにする必要があります。システム固有の z/OS UNIX ファイル・システムを指定のパスにマウントする場合は、rsed.envvars 内の daemon.log および user.log ディレクティブを使用して z/OS UNIX のログを特

定のロケーションに送付することで、構成ファイルを共用できます。こうすると、すべてのログが論理的には同じ場所書き込まれますが、その下のファイル・システムが共用されていないので、物理的には別々のロケーションに置かれることになります。

- /etc/rdz/ や /var/rdz/projects/ のような構成タイプのディレクトリーは、Developer for System z が読み取り専用モードで使用するため、シスプレックス全体で共用できます。
- 一時ファイル名はシスプレックスで認識されないため、/tmp/ や /var/rdz/WORKAREA/ のような一時データ・ディレクトリーは、システムごとに固有である必要があります。
- コードを共用する場合は、保守適用後に同期のとれていないシステムが残らないように、構成ファイルも共用する必要があります。

同一のソフトウェア・レベル、異なる構成ファイル

限定された一連の環境においては、カスタマイズ可能な部分 (の一部) を除くすべてを共用できます。1 つの例は、オンサイト使用のために非 SSL アクセスを提供し、オフサイト使用のために SSL エンコード通信を提供することです。

重要: 共用セットアップは、保守、テクニカル・プレビュー、または新規リリースのテストには、安全に使用できません。

アクティブな Developer for System z インストール済み環境の別のインスタンスをセットアップするには、現行セットアップとのオーバーラップを避けるため、異なるデータ・セット、ディレクトリー、およびポートを使用して、異なる部分のカスタマイズ・ステップを再実行します。

上記の SSL サンプルでは、現行 RSE デーモンのセットアップのクローンを作成でき、その後、クローンのセットアップを更新できます。次に、RSE デーモン始動 JCL のクローンを作成し、新しい TCP/IP ポートと更新した構成ファイルのロケーションを使用してカスタマイズすることができます。MVS カスタマイズ (JES ジョブ・モニターなど) は、SSL インスタンスと非 SSL インスタンスの間で共用できます。結果として、アクションは以下のようになります。

```
$ cd /etc/rdz
$ mkdir /etc/rdz/ssl
$ cp rsed.envvars /etc/rdz/ssl
$ cp ssl.properties /etc/rdz/ssl
$ ls /etc/rdz/ssl/
rsed.envvars    ssl.properties
$ oedit /etc/rdz/ssl/rsed.envvars
-> uncomment and change: -Ddaemon.log=/var/rdz/logs/ssl
-> add at the END:
# -- NEEDED TO FIND THE REMAINING CONFIGURATION FILES
CFG_BASE=/etc/rdz
CLASSPATH=.:$CFG_BASE:$CLASSPATH
# --
$ oedit /etc/rdz/ssl/ssl.properties
-> change: change as needed
```

上記のコマンドは、変更が必要な Developer for System z 構成ファイルを、新規に作成された ssl ディレクトリーにコピーします。rsed.envvars 内の daemon.log 変数を更新して、新しいログ・ロケーション (これは、存在しなければ自動的に作

成されます) を定義する必要があります。CLASSPATH の更新は、ssl にコピーされなかった構成ファイルを RSE が確実に検出できるようにします。ssl.properties ファイル自体を、必要性に合わせて更新できます。

この時点で残っている作業は、新しいポート番号と新しい /etc/rdz/ssl 構成ファイルを使用する RSE 用の新規開始タスクを作成することです。

上記の各アクションの詳細については、この資料の関連するセクションを参照してください。

その他のすべての状態

コードの変更が関与する場合 (保守、テクニカル・プレビュー、新規リリース)、または変更がかなり複雑な場合は、Developer for System z の別のインストールを実行する必要があります。ここでは、さまざまなインストール間で発生する可能性がある競合点について説明します。

次のリストは、Developer for System z の複数のインスタンス間で異なるものにする必要があるか、そうすることが強く推奨される項目の要約です。

- SMP/E CSI
- インストール・ライブラリー
- JES ジョブ・モニター TCP/IP ポート、およびその構成ファイル FEJJCNFG
- JES ジョブ・モニター始動 JCL
- APPC トランザクション名
- RSE 構成ファイル、rsed.envvars、*.properties、および *.settings
- RSE TCP/IP ポート
- RSE 始動 JCL

以下で、これらの概要を説明します。

- SMP/E CSI
 1. Developer for System z の各インスタンスを別々の CSI にインストールします。SMP/E は同じ FMID が 1 つの CSI に再度インストールされるのを阻止しますが、別の FMID のインストールは受け入れます。2 番目の FMID の方が新しいバージョンであれば、既存のバージョンの製品は削除されます。2 番目の FMID の方が古いバージョンであれば、インストールはパーツ名の重複のために失敗します。
- インストール・ライブラリー
 1. Developer for System z の各インスタンスを別々のデータ・セットおよびディレクトリーにインストールします。留意すべき点は、IBM 提供のデフォルトの /usr/lpp/rdz にプレフィックスを付けることによって、z/OS UNIX パスだけを変更できるということです。妥当な例は、/service/usr/lpp/rdz です。
 2. カスタマイズ・セットアップ・ジョブ FEK.SFEKSAMP(FEKSETUP) は、構成ファイルを保管するために使用するデータ・セットとディレクトリーを作成します。構成ファイルは固有でなければならず、既存のカスタマイズを上書きしな

いようにするために、このジョブを実行依頼するときは、固有のデータ・セット名とディレクトリー名を使用する必要があります。

• 必須部分

1. JES ジョブ・モニター構成ファイル FEK.#CUST.PARMLIB(FEJJCNFG) は、JES ジョブ・モニターの TCP/IP ポート番号を保持しているので、共用することはできません。このメンバー自体は名前を変更できるので (JCL も更新する場合)、インストール・データ・セット内で更新を行うのでない場合は、このメンバーのカスタマイズ済みバージョンをすべて、同じデータ・セット内に配置できます。
2. JES ジョブ・モニター始動 JCL FEK.#CUST.PROCLIB(JMON) は FEJJCNFG を参照し、したがって、これも共用することはできません。メンバー (および、ユーザー・ジョブとして始動する場合は JOB カード) を名前変更した後、すべての JCL を同じデータ・セット内に配置できます。
3. RSE 構成ファイル /etc/rdz/rsed.envvars は、インストール・パスへの参照を保持し、オプションとして、サーバー・ログ・ロケーション (これは固有であることが必要です) への参照も保持します。このファイル名は必須なので、同じディレクトリー内にさまざまなコピーを保持することはできません。
4. ISPF.conf 構成ファイルは FEK.SFEKPROC(FEKFRSRV)、つまり TSO コマンド・サーバーを参照しています。これはソフトウェア・レベル固有なので、インスタンスごとに ISPF.conf ファイルを作成する必要があります。
5. 上記以外のすべての z/OS UNIX ベースの構成ファイル (*.properties など) は、rsed.envvars と同じディレクトリーに存在する必要があるため、したがって、共用できません。rsed.envvars は非共用ロケーションに置かれている必要があるからです。
6. RSE 始動 JCL の FEK.#CUST.PROCLIB(RSED) は TCP/IP ポート番号を定義しており、固有でなければならないインストールおよび構成ディレクトリーを参照しているため、共用できません。メンバー (および、ユーザー・ジョブとして始動する場合は JOB カード) を名前変更した後、すべての JCL を同じデータ・セット内に配置できます。
7. ロック・デーモン始動 JCL FEK.#CUST.PROCLIB(LOCKD) は、固有でなければならないインストールおよび構成ディレクトリーを参照しているため、共用できません。メンバー (およびユーザー・ジョブとして始動する場合は JOB カード) を名前変更した後、すべての JCL を同じデータ・セット内に配置できます。

• オプションのパーツ

1. REXEC ポートおよび SSH TCP/IP ポートは、無制限に共用できます。
2. APPC トランザクションは FEK.SFEKPROC(FEKFRSRV)、つまり TSO コマンド・サーバーを参照しています。これはソフトウェア・レベル固有なので、インスタンスごとに APPC トランザクションを作成する必要があります。APPC トランザクション名は変化するので、rsed.envvars 内で _FEKFSCMD_TP_NAME_ 変数を定義する必要があることに留意してください。
3. 一部の ELAXF* プロシージャは hlq.SFEKLOAD、つまり Developer for System z のロード・ライブラリーを参照しています。ユーザーが使用できるさまざまなセットの作成方法については、27 ページの『ELAXF* リモート・ビルド・プロシージャ』で JCLLIB についての注を参照してください。

4. DB2 ストアド・プロシージャの 2 つのインスタンスをアクティブにするには、以下の作業を完了する必要があります。ただし、この説明はサポートの対象ではなく、保証がないことに注意してください。

- a. hlq.SFEKPROC(ELAXMREX) を、別の名前を付けたメンバー、例えば ELAXMRXX にコピーします。
- b. サンプル・メンバー hlq.SFEKSAMP(ELAXMSAM) を、別の名前を付けたメンバー、例えば ELAXMWDZ にコピーします。
- c. これらの名前変更が反映されるように、サンプル・メンバー hlq.SFEKSAMP(ELAXMJCL) を変更します。以下に例を示します。

```
//SYSIN DD *
CREATE PROCEDURE SYSPROC.ELAXMRXX
( IN FUNCTION_REQUEST VARCHA(20) CCSID EBCDIC
...
, OUT RETURN_VALUE VARCHA(255) CCSID EBCDIC )
PARAMETER STYLE GENERAL RESULT SETS 1
LANGUAGE REXX EXTERNAL NAME ELAXMRXX
COLLID DSNREXCS WLM ENVIRONMENT ELAXMWDZ
PROGRAM TYPE MAIN MODIFIES SQL DATA
STAY RESIDENT NO COMMIT ON RETURN NO
ASUTIME NO LIMIT SECURITY USER;

COMMENT ON PROCEDURE SYSPROC.ELAXMRXX IS
'PLI & COBOL PROCEDURE PROCESSOR (ELAXMRXX), INTERFACE LEVEL 0.01';

GRANT EXECUTE ON PROCEDURE SYSPROC.ELAXMRXX TO PUBLIC;
//
```

- d. 95 ページの『(オプション) DB2 ストアド・プロシージャ』の説明に従って (ただし、新しいメンバーを使用して) カスタマイズを続行します。
 - e. 新しい WLM 環境名 (例えば、ELAXMWDZ) を、クライアント上の DB2 ストアド・プロシージャ・ウィザードで使用する必要があります。
5. CICS 領域での Bidi サポートはロード・ライブラリー・メンバーに依存しているため、複数のリリースにまたがって共用することはできません。しかし、そのロード・モジュール名がすべてのインスタンスで同一の場合は、複数のリリースにまたがっても最新バージョンをインスタンス間で共用できます。ロード・モジュールの名前が変更された場合は、後方互換性を利用できません。
6. CICS 領域に組み込まれている Application Deployment Manager ロード・モジュールには後方互換性があるので、最新バージョンを複数のリリースにまたがって共用できます。
7. Application Deployment Manager の CRD VSAM には後方互換性があるので、最新バージョンを複数のリリースにまたがって共用できます。
8. Application Deployment Manager の CICS リソース定義には後方互換性があるので、最新バージョンを複数のリリースにまたがって共用できます。
9. CARMA VSAM はソフトウェア・レベル間で変更されている可能性があるため、共用はお勧めできません。

第 18 章 マイグレーション・ガイド

マイグレーションに関する考慮事項

ここでは、本製品の以前のリリースと比較したインストールおよび構成上の変更点に重点を置いて説明します。また、このリリースへのマイグレーションに関する一般的なガイドラインも示します。詳細については、本書内の関連するセクションを参照してください。

- 以前に IBM Rational Developer for System z、IBM WebSphere Developer for System z、IBM WebSphere Developer for zSeries、または IBM WebSphere® Studio Enterprise Developer を使用していた場合は、IBM Rational Developer for System z バージョン 7.6.1 へのアップグレードをインストールする「前」に、関連するカスタマイズ済みファイルを保存しておくことをお勧めします。
- Developer for System z の複数のインスタンスを実行する計画の場合は、291 ページの『第 17 章 複数のインスタンスの実行』を参照してください。

注: ここに示すマイグレーション情報は、この資料の公開時にまだサポートされている Developer for System z のバージョンに関するものです。

前に構成したファイルのバックアップ

以前に Developer for System z を使用していた場合は、このバージョンの IBM Developer for System z をインストールする前に、関連するカスタマイズ済みファイルを保存しておくことをお勧めします。

カスタマイズ可能な Developer for system z ファイルは、以下のロケーションにあります。

- バージョン 7.5
 - FEK.SFEKSAMP。一部のメンバーは、FEKSETUP サンプル・ジョブによって FEK.#CUST.* にコピーされます。ここで、* は PARMLIB、PROCLIB、JCL、CNTL、ASM、および COBOL に相当します。
 - FEK.SFEKSAMV
 - /usr/lpp/rdz/samples/。一部のファイルは、FEKSETUP サンプル・ジョブによって /etc/rdz/ および /etc/rdz/sc1mdt/* にコピーされます。ここで、* は CONFIG/、CONFIG/PROJECT/、および CONFIG/script/ に相当します。
- バージョン 7.1
 - FEK.SFEKSAMP
 - CRA.SCRASAMP
 - /usr/lpp/wd4z/rse/lib/。カスタマイズ可能なファイルは /etc/wd4z/ にコピーすることをお勧めします。
- バージョン 7.0
 - FEK.SFEKSAMP
 - CRA.SCRASAMP

- /usr/lpp/wd4z/rse/lib/。カスタマイズ可能なファイルは /etc/wd4z/ にコピーすることをお勧めします。

以前の Developer for system z セットアップでは、他の製品によって所有されている構成ファイルへの変更も文書化されます。

- バージョン 7.5

- SYS1.PARMLIB(ASCHPMxx)

TSO コマンド・サービス用の APPC トランザクション・クラスを定義します。

- SYS1.PARMLIB(BPXPRMxx)

z/OS UNIX システムのデフォルトを設定します。

- SYS1.PARMLIB(COMMNDxx)

IPL 時にサーバーを始動します。

- SYS1.PARMLIB(LPALSTxx)

FEK.SFEKLPA を LPA に追加します。

- SYS1.PARMLIB(PROGxx)

FEK.SFEKAUTH の APF 許可

FEK.SFEKAUTH および FEK.SFEKLOAD を LINKLIST に追加します。

- (APPC)

TSO コマンド・サービス用の APPC トランザクションを定義します。

- (WLM)

APPC トランザクション・プログラムを TSO パフォーマンス・グループに関連付けます。

- (WLM)

DB2 ストアード・プロシージャのアプリケーション環境を割り当てます。

- バージョン 7.1

- SYS1.PARMLIB(ASCHPMxx)

TSO コマンド・サービス用の APPC トランザクション・クラスを定義します。

- SYS1.PARMLIB(BPXPRMxx)

z/OS UNIX システムのデフォルトを設定します。

- SYS1.PARMLIB(PROGxx)

FEK.SFEKLOADの APF 許可

- /etc/services

RSE デーモン・ポートを定義します。

- /etc/inetd.conf

- RSE デモン・サービスを定義します。
- /etc/SCLMDT/CONFIG/ISPF.conf
- TSO コマンド・サーバーのロケーションを定義します。
- (APPC)
- TSO コマンド・サービス用の APPC トランザクションを定義します。
- (WLM)
- APPC トランザクション・プログラムを TSO パフォーマンス・グループに関連付けます。
- (WLM)
- DB2 ストアード・プロシージャのアプリケーション環境を割り当てます。
- バージョン 7.0
 - SYS1.PARMLIB(ASCHPMxx)
 - TSO コマンド・サービス用の APPC トランザクション・クラスを定義します。
 - SYS1.PARMLIB(BPXPRMxx)
 - z/OS UNIX システムのデフォルトを設定します。
 - SYS1.PARMLIB(PROGxx)
 - FEK.SFEKLOAD の APF 許可
 - /etc/services
 - RSE デモン・ポートを定義します。
 - /etc/inetd.conf
 - RSE デモン・サービスを定義します。
 - (APPC)
 - TSO コマンド・サービス用の APPC トランザクションを定義します。
 - (WLM)
 - APPC トランザクション・プログラムを TSO パフォーマンス・グループに関連付けます。
 - (WLM)
 - DB2 ストアード・プロシージャのアプリケーション環境を割り当てます。

バージョン 7.6.1 のマイグレーションに関する注

- 以下に示すマイグレーションに関する注は、バージョン 7.6.1 に固有です。これらの注は、バージョン 7.6 からのマイグレーションに有効であるか、またはバージョン 7.6 のマイグレーションに関する注への追加です。
- Application Deployment Manager - CICS RPL 連結内の既存の ADN* モジュールを更新する必要があります。

- Application Deployment Manager - 管理ユーティリティでの URIMAP サポートを追加するために、以下のサンプル・メンバーが更新されました。
 - ADNJSPAU
 - ADNVCRD
- Application Deployment Manager - URIMAP サポートを使用可能にするには、既存の CRD リポジトリ VSAM を置き換える必要があります。
- CARMA - CARMA カスタム情報 VSAM データ・セット CRASTRS の可変長レイアウトのサポートが追加されました。
- CARMA - 新しいサンプル・メンバーが追加されました。
 - CRA#VS2 - CRASTRS を可変長フォーマットにマイグレーションします。
- JES ジョブ・モニター - 開始タスク JCL での _CEE_ENVFILE_S の使用。
- JES ジョブ・モニター - 以下の FEJJCNFG ディレクティブがオプションになりました。
 - HOST_CODEPAGE
- PROCLIB - 新しい PROCLIB メンバーが追加されました。
 - ELAXFDCL
- RSE - 64 ビット Java の使用がサポートされるようになりました。
- RSE - 新しいオペレーター・コマンドが追加されました (バージョン 7.6.1.0 より)。
 - MODIFY DISPLAY PROCESS,DETAIL
- RSE - rsed.envvars で、以下のカスタマイズ不可能なディレクティブが変更または新たに追加されました (バージョン 7.6.0.0 より)。
 - (_RSE_JAVAOPTS) -DDSTORE_KEEPALIVE_RESPONSE_TIMEOUT
 - (_RSE_JAVAOPTS) -DDSTORE_IO_SOCKET_READ_TIMEOUT
 - (_RSE_JAVAOPTS) -DRSECOMM_LOGFILE_MAX
- RSE - rsed.envvars に新しいオプションのディレクティブが追加されました (バージョン 7.6.0.0 および 7.6.0.1 より)。
 - (_RSE_JAVAOPTS) -Denable.automount
 - (_RSE_JAVAOPTS) -Ddeny.nozero.port
 - (_RSE_JAVAOPTS) -Dsingle.logon
 - (_RSE_JAVAOPTS) -Dprocess.cleanup.interval
- RSE - 以下のコンソール・メッセージが変更または新たに追加されました (バージョン 7.6.0.1 および 7.6.1.0 より)。
 - FEK001I
 - FEK210I

バージョン 7.5 からバージョン 7.6 へのマイグレーション

IBM Rational Developer for System z、FMID HHOP760

- SMP/E による MVS および z/OS UNIX コンポーネントのデフォルトのインストール・ロケーションは、変更されておらず、したがって FEK.* および /usr/lpp/rdz/* のままです。
- Application Deployment Manager - CICS RPL 連結内の既存の ADN* モジュールを更新する必要があります。
- Application Deployment Manager - CICS RESTful インターフェースをサポートするために、新しいロード・モジュール (これは CICS RPL 連結の一部でなければなりません) が追加されました。
 - ADNANAL
 - ADNCRD41
 - ADNREST
- Application Deployment Manager - CICS RESTful インターフェースをサポートするために、新しいサンプル・メンバーが追加されました。
 - ADNCSDRS
 - ADNCSDTX
 - ADNTXNC
- Application Deployment Manager - 既存のサンプル・メンバーの名前が変更されました。
 - ADNARCSO -> ADNCSDAR
 - ADNCMSGH -> ADNMSGHC
 - ADNMFEST -> ADNVMFST
 - ADNPCCSD -> ADNCSDWS
 - ADNSMSGH -> ADNMSGHS
 - ADNVSAM -> ADNVCRD
- CA Endeavor® SCM にアクセスするための新しい実動タイプの RAM が提供されています。
 - CRARNDVR
- CARMA - CA Endeavor® SCM RAM をサポートするために、新しいサンプル・メンバーが提供されています。
 - FEK.#CUST.JCL(CRA#VCAD)
 - FEK.#CUST.JCL(CRA#VCAS)
 - FEK.#CUST.CNTL(CRASUBCA)
 - FEK.#CUST.PARMLIB(CRASHOW)
 - FEK.#CUST.PARMLIB(CRATMAP)
 - FEK.SFEKPROC(CRANDVRA)
 - /etc/rdz/crastart.endevor.conf
- CARMA - RAM 定義のマージをサポートするために、新しいサンプル・メンバーが提供されています。

- CRA#UADD
- CRA#UQRY
- File Manager Integration - File Manager にアクセスするためのバッチ・インターフェースは、サポートされなくなりました。
- File Manager Integration - FMIEXT.properties 構成ファイルが全面的に変更されたため、このファイルを置き換える必要があります。
- JES ジョブ・モニター - LE オプションが FEJJMON ロード・モジュールに (バージョン 7.5.0.1 以降) 埋め込まれました。これにより、ご使用の開始タスク定義に変更が必要になる場合があります。詳しくは、FEK.SFEKSAMP(FEJJJCL) サンプル JCL を参照してください。
- JES ジョブ・モニター - 新しいオプションのディレクティブが (バージョン 7.5.0.1 および 7.5.1.0) の FEJJCNGF に追加されました。
 - APPLID
 - CONSOLE_NAME
 - GEN_CONSOLE_NAME
- JES ジョブ・モニター - 新しいコマンドの「JCL の表示」が (バージョン 7.5.1.0 以降) サポートされています。このため、場合によっては、ご使用のセキュリティ・ソフトウェアを更新する必要があります。
- ロック・デーモン - ロック・デーモン (LOCKD) は (バージョン 7.5.0.1 以降の) 新しい開始タスクです。この開始タスクに照会して、どの Developer for z クライアントがデータ・セット・ロックを保持しているかを識別できます。(システム・コマンドは、RSE スレッド・プールであるアドレス・スペース・レベルで停止します。)
- SCLMDT - SCLMDT プロジェクト構成ファイルのデフォルト・ロケーションが変更されました。
 - /var/rdz/sclmdt
- RSE - 新しいオペレーター・コマンドが追加されました。
 - MODIFY RSESTANDARDLOG
- RSE - 新しい必須ディレクティブが (バージョン 7.5.0.1 および 7.6.0.0 で) rsed.envvars に追加されました。
 - _RSE_LOCKD_PORT
 - (_RSE_JAVAOPTS) -Dlock.daemon.port
 - (_RSE_JAVAOPTS) -Dlock.daemon.cleanup.interval
 - _RSE_LOCKD_CLASS
 - _RSE_HOST_CODEPAGE
 - (_RSE_JAVAOPTS) -Dfile.encoding
 - (_RSE_JAVAOPTS) -Dconsole.encoding
- RSE - 新しいオプションのディレクティブが (バージョン 7.5.0.1、7.5.1.0、および 7.6.0.0 で) rsed.envvars に追加されました。
 - (_RSE_JAVAOPTS) -Duser.log
 - (_RSE_JAVAOPTS) -Dkeep.last.log
 - (_RSE_JAVAOPTS) -Denable.standard.log

- (_RSE_JAVAOPTS) -DDSTORE_LOG_DIRECTORY
- (_RSE_JAVAOPTS) -Dhide_ZOS_UNIX
- (_RSE_JAVAOPTS) -Denable.certificate.mapping
- GSK_CRL_SECURITY_LEVEL
- GSK_LDAP_SERVER
- GSK_LDAP_PORT
- GSK_LDAP_USER
- GSK_LDAP_PASSWORD
- RSE - 一部のオプションのディレクティブが `rsed.envvars` で変更されました。
 - (_RSE_JAVAOPTS) -Ddaemon.log
 - (_RSE_JAVAOPTS) -Xmx
 - SCLMDT_CONF_HOME
- RSE - 新しいオプションのディレクティブが (バージョン 7.5.1.0 および 7.6.0.0 以降で) `ssl.properties` に追加されました。
 - `server_keystore_label`
 - `server_keystore_type`
- RSE - RSE デーモンは X.509 クライアント証明書認証を (バージョン 7.5.1.0 以降で) サポートしています。このため、現行の証明書とセキュリティ・セットアップ (使用している場合) の更新が必要になります。
- RSE - セキュリティーが強化され、PassTicket および FEKAPPL のエラーが発生すると接続要求が失敗するようになりました。
- RSE - すべてのログ・ファイル (デーモン・ログとユーザー・ログ) のデフォルト・ロケーションが変更されました。
 - `/var/rdz/logs`
 - `/var/rdz/logs/$LOGNAME`
- RSE - Developer for System z のログと構成情報を収集するための新しいサンプル JCL が提供されています。
 - FEKLOGS

構成可能なファイル

304 ページの表 47 は、バージョン 7.6 でカスタマイズされるファイルの概要を示しています。Developer for System z のサンプル・ライブラリー FEK.SFEKSAMP、FEK.SFEKSAMV、および `/usr/lpp/rdz/samples/` は、ここにリストしたものより多くのカスタマイズ可能なメンバーが付属する場合があることに注意してください (サンプルの CARMA ソース・コードおよびそれらをコンパイルするジョブなど)。

注: サンプル・ジョブ FEKSETUP は、リストされているすべてのメンバーを別のデータ・セットおよびディレクトリー (デフォルトでは `FEK.#CUST.*` および `/etc/rdz/*`) にコピーします。

表 47. バージョン 7.6 のカスタマイズ

メンバー/ファイル	デフォルト・ロケーション	目的	マイグレーションの注
FEKSETUP	FEK.SFEKSAMP [FEK.#CUST.JCL]	データ・セットおよびディレクトリーを作成し、カスタマイズ可能ファイルのデータを取り込むための JCL	新しいカスタマイズ可能なメンバーを追加するために更新された
JMON	FEK.SFEKSAMP (FEJJJCL) [FEK.#CUST.PROCLIB]	JES ジョブ・モニター用の JCL	LE オプションを変更するために追加されたオプション
FEJJJCL	FEK.SFEKSAMP [FEK.#CUST.PROCLIB (JMON)]	JMON メンバーの出荷名	JMON メンバーを参照
RSED	FEK.SFEKSAMP (FEKRSED) [FEK.#CUST.PROCLIB]	RSE デーモンの JCL	なし
FEKRSED	FEK.SFEKSAMP [FEK.#CUST.PROCLIB (RSED)]	RSED メンバーの出荷名	RSED メンバーを参照
LOCKD	FEK.SFEKSAMP (FEKLOCKD) [FEK.#CUST.PROCLIB]	ロック・デーモン用の JCL	新規。カスタマイズが必要
FEKLOCKD	FEK.SFEKSAMP [FEK.#CUST.PROCLIB (LOCKD)]	LOCKD メンバーの出荷名	LOCKD メンバーを参照
ELAXF*	FEK.SFEKSAMP [FEK.#CUST.PROCLIB]	リモート・プロジェクト・ビルドなどのための JCL	なし
FEKRACF	FEK.SFEKSAMP [FEK.#CUST.JCL]	セキュリティ定義の JCL	マイナー更新
FEJJCNFG	FEK.SFEKSAMP [FEK.#CUST.PARMLIB]	JES ジョブ・モニター構成ファイル	オプションの新規ディレクティブが追加された
FEJTSO	FEK.SFEKSAMP [FEK.#CUST.CNTL]	TSO 実行依頼用の JCL	なし
CRA\$VMSG	FEK.SFEKSAMP [FEK.#CUST.JCL]	CARMA メッセージ VSAM を作成するための JCL	なし
CRA\$VDEF	FEK.SFEKSAMP [FEK.#CUST.JCL]	CARMA 構成 VSAM を作成するための JCL	なし
CRA\$VSTR	FEK.SFEKSAMP [FEK.#CUST.JCL]	CARMA カスタム情報 VSAM を作成するための JCL	なし
CRASUBMT	FEK.SFEKSAMP [FEK.#CUST.CNTL]	CARMA バッチ始動 CLIST	なし
CRASUBCA	FEK.SFEKSAMP [FEK.#CUST.CNTL]	CA Endeavor® SCM RAM 用の CARMA バッチ始動 CLIST	新規、カスタマイズはオプション
CRASHOW	FEK.SFEKSAMP [FEK.#CUST.PARMLIB]	CA Endeavor® SCM RAM 用の CARMA 構成	新規、カスタマイズはオプション
CRATMAP	FEK.SFEKSAMP [FEK.#CUST.PARMLIB]	CA Endeavor® SCM RAM 用の CARMA 構成	新規、カスタマイズはオプション

表 47. バージョン 7.6 のカスタマイズ (続き)

メンバー/ファイル	デフォルト・ロケーション	目的	マイグレーションの注
CRANDVRA	FEK.SFEKPROC	CA Endeavor® SCM RAM 用の CARMA 割り振り REXX	新規、カスタマイズはオプション
CRAISPRX	FEK.SFEKSAMP [FEK.#CUST.CNTL]	TSO/ISPF クライアント・ゲートウェイを使用した CARMA 用のサンプル DD 割り振り exec	なし
CRA#VSLM	FEK.SFEKSAMP [FEK.#CUST.JCL]	SCLM RAM のメッセージ VSAM を作成するための JCL	なし
CRA#ASLM	FEK.SFEKSAMP [FEK.#CUST.JCL]	SCLM RAM のデータ・セットを作成するための JCL	なし
CRA#VPDS	FEK.SFEKSAMP [FEK.#CUST.JCL]	PDS RAM のメッセージ VSAM を作成するための JCL	なし
CRA#CRAM	FEK.SFEKSAMP [FEK.#CUST.JCL]	スケルトン RAM をコンパイルするための JCL	なし
CRA#VCAD	FEK.SFEKSAMP [FEK.#CUST.JCL]	CA Endeavor® SCM RAM 用に CARMA 構成 VSAM を作成するための JCL	新規、カスタマイズはオプション
CRA#VCAS	FEK.SFEKSAMP [FEK.#CUST.JCL]	CA Endeavor® SCM RAM 用に CARMA カスタム情報 VSAM を作成するための JCL	新規、カスタマイズはオプション
CRA#UADD	FEK.SFEKSAMP [FEK.#CUST.JCL]	RAM 定義をマージするための JCL	新規、カスタマイズはオプション
CRA#UQRY	FEK.SFEKSAMP [FEK.#CUST.JCL]	RAM 定義を抽出するための JCL	新規、カスタマイズはオプション
CRAXJCL	FEK.SFEKSAMP [FEK.#CUST.ASM]	IRXJCL 置換用のサンプル・ソース・コード	なし
CRA#CIRX	FEK.SFEKSAMP [FEK.#CUST.JCL]	CRAXJCL をコンパイルするための JCL	なし
ADNCSDRS	FEK.SFEKSAMP [FEK.#CUST.JCL]	RESTful CRD サーバーを主 CICS 領域に対して定義するための JCL	新規、カスタマイズはオプション
ADNCSDTX	FEK.SFEKSAMP [FEK.#CUST.JCL]	代替トランザクション ID を CICS 領域に対して定義するための JCL	新規、カスタマイズはオプション
ADNTXNC	FEK.SFEKSAMP [FEK.#CUST.JCL]	代替トランザクション ID を作成するための JCL	新規、カスタマイズはオプション
ADNMSGHC	FEK.SFEKSAMP [FEK.#CUST.JCL]	ADNMSGHS をコンパイルするための JCL	名前変更。旧 ADNMSGH

表 47. バージョン 7.6 のカスタマイズ (続き)

メンバー/ファイル	デフォルト・ロケーション	目的	マイグレーションの注
ADNMSGHS	FEK.SFEKSAMP [FEK.#CUST.COBOL]	パイプライン・メッセージ・ハンドラー用のサンプル・ソース・コード	名前変更。旧 ADNMSGH
ADNVCRD	FEK.SFEKSAMP [FEK.#CUST.JCL]	CRD リポジトリを作成するための JCL	名前変更。旧 ADNVSAM
ADNCSDWS	FEK.SFEKSAMP [FEK.#CUST.JCL]	Web サービス CRD サーバーを主 CICS 領域に対して定義するための JCL	名前変更。旧 ADNPCCSD
ADNCSDAR	FEK.SFEKSAMP [FEK.#CUST.JCL]	CRD サーバーを非主 CICS 領域に対して定義するための JCL	名前変更。旧 ADNARCS
ADNJSPAU	FEK.SFEKSAMP [FEK.#CUST.JCL]	CRD のデフォルトを更新するための JCL	RESTful サービス用の定義が追加されたため、カスタマイズを再実行する必要があります。
ADNMFST	FEK.SFEKSAMP [FEK.#CUST.JCL]	マニフェスト・リポジトリを作成し、定義するための JCL	名前変更。旧 ADNMFEST
ELAXMSAM	FEK.SFEKSAMP [FEK.#CUST.PROCLIB]	PL/I および COBOL ストアード・プロシージャ・ビルダー用の WLM アドレス・スペースの JCL プロシージャ	なし
ELAXMJCL	FEK.SFEKSAMP [FEK.#CUST.JCL]	PL/I および COBOL ストアード・プロシージャ・ビルダーを DB2 に対して定義するための JCL	なし
FEKAPPCC	FEK.SFEKSAMP [FEK.#CUST.JCL]	APPC トランザクションを作成するための JCL	なし
FEKAPPCL	FEK.SFEKSAMP [FEK.#CUST.JCL]	APPC トランザクションを表示するための JCL	なし
FEKAPPX	FEK.SFEKSAMP [FEK.#CUST.JCL]	APPC トランザクションを削除するための JCL	なし
FEKLOGS	FEK.SFEKSAMP [FEK.#CUST.JCL]	ログ・ファイルを収集するための JCL	新規、カスタマイズはオプション
rsed.envvars	/usr/lpp/rdz/samples/ [/etc/rdz/]	RSE 環境変数	古いコピーは、このコピーに置き換える必要がある (カスタマイズの再実行が必要)

表 47. バージョン 7.6 のカスタマイズ (続き)

メンバー/ファイル	デフォルト・ロケーション	目的	マイグレーションの注
ISPF.conf	/usr/lpp/rdz/samples/ [/etc/rdz/]	TSO/ISPF クライアント・ゲートウェイ構成ファイル	SCLMDT 用に SYSPROC に追加された ISP.SISPCLIB
CRASRV.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	CARMA 構成ファイル	なし
crastart.conf	/usr/lpp/rdz/samples/ [/etc/rdz/]	CRASTART を使用するための CARMA 構成ファイル	なし
crastart.endevor.conf	/usr/lpp/rdz/samples/ [/etc/rdz/]	CA Endevor® SCM RAM 用に CRASTART を使用するための CARMA 構成ファイル	新規、カスタマイズはオプション
ssl.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	RSE SSL 構成ファイル	オプションの新規ディレクティブが追加された
rsecomm.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	RSE トレース構成ファイル	なし
propertiescfg.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	ホスト・ベース・プロパティ・グループ構成ファイル	なし
projectcfg.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	ホスト・ベース・プロジェクト構成ファイル	なし
FMEXT.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	File Manager Integration 構成ファイル	古いコピーは、このコピーに置き換える必要がある (カスタマイズの再実行が必要)
uchars.settings	/usr/lpp/rdz/samples/ [/etc/rdz/]	編集不可能文字構成ファイル	なし

バージョン 7.1 からバージョン 7.5 へのマイグレーション

IBM Rational Developer for System z、FMID HHOP750

- SMP/E による MVS コンポーネントのデフォルトのインストール・ロケーションは、変更されておらず、したがって FEK.* のままです。
- SMP/E による z/OS UNIX コンポーネントのデフォルトのインストール・ロケーションは、/usr/lpp/rdz/* に変更されました。
- 共通アクセス・リポジトリ・マネージャー (CARMA) は Developer for System z バージョン 7.5 にマージされたため、別個の製品としてインストールする必要がなくなりました。

- SCLM Developer Toolkit は Developer for System z バージョン 7.5 にマージされたため、別個の製品としてインストールする必要がなくなりました。
- バージョン 7.5 では、ISPF の TSO/ISPF クライアント・ゲートウェイ・サービスは、バージョン 7.1 で使用されていた SCLM Developer Toolkit 機能に代わって TSO コマンド・サービスに接続します。APPC 接続方式は引き続きサポートされ、機能します。
- バージョン 7.5 では、RSE サーバーは INETD 管理対象プロセスではなく、開始タスクになりました。RSE サーバーは、現在は単一サーバー・モデルも使用します。これに対し、以前のバージョンでは、それぞれのクライアント/ホスト接続ごとに専用の RSE サーバーがありました。
- APF 許可を必要とするすべてのモジュール (JES ジョブ・モニターおよび SCLM Developer Toolkit) が、バージョン 7.5 で FEK.SFEKAUTH に移動したため、既存の APF 定義を更新する必要があります。
- JES ジョブ・モニターのロード・モジュールが、バージョン 7.5 で FEK.SFEKAUTH に移動したため、既存の開始タスク・プロシーチャーを更新する必要があります。
- CARMA ロード・モジュールが新しいライブラリーに移動したため、既存の CRASUBMT サーバー始動スクリプトを更新する必要があります。
- SCLM Developer Toolkit ロード・モジュールが新しいライブラリーに移動したため、既存の LINKLIST 定義を更新する必要があります。
- ELAXFTSO は、バージョン 7.1.1 以降の新しいサンプル・ビルド・プロシーチャーで、ELAXFCP1 および ELAXFPP1 は、バージョン 7.5 の新しいプロシーチャーです。
- uchars.settings は、編集不可能文字の新しい構成ファイルです。
- propertiescfg.properties はデフォルトのプロパティー・グループの新しい構成ファイルです。
- FEJJCNFG、CRASRV.properties、および FMIEXT.properties には、新しいオプションのディレクティブがあります。
- rsed.envvars は、バージョン 7.5 で変更されたため、置き換える必要があります。
- バージョン 7.5 に添付されているサンプル ISPF.conf ファイルは、バージョン 7.1 で SCLM Developer Toolkit が使用していたものとよく似ています。
- Application Deployment Manager の既存のカスタマイズの一部は、再実行する必要があります。
- Application Deployment Manager には、カスタマイズが必要な新しい機能があります。
- RSE サーバーのセキュリティー設定は、バージョン 7.5 で大幅に変更されました。
- MVS.MCSOPER.JMON セキュリティー・プロファイルは、バージョン 7.5 の JES ジョブ・モニターの新機能です。
- CARMA 始動スクリプトの名前が変更され、新しいロケーションに移動したため、既存の CRASRV.properties 構成ファイルを更新する必要があります。
- FMI 始動スクリプトの名前が変更され、新しいロケーションに移動したため、既存の FMIEXT.properties 構成ファイルを更新する必要があります。

- 双方向言語サポート用に新しいモジュールが追加されたため、FEK.SFEKLOAD ライブラリーを使用していない場合は、既存の CICS DFHRPL 連結を更新する必要があります。
- SYS1.PARMLIB(BPXPRMxx) の MAXPROCUSER パラメーターの変更について、文書化も行われました。

構成可能なファイル

表 48 は、バージョン 7.5 でカスタマイズされるファイルの概要を示しています。Developer for System z のサンプル・ライブラリー FEK.SFEKSAMP、FEK.SFEKSAMV、および /usr/lpp/rdz/samples/ は、ここにリストしたものより多くのカスタマイズ可能なメンバーが付属する場合がありますことに注意してください (サンプルの CARMA ソース・コードおよびそれらをコンパイルするジョブなど)。

注: サンプル・ジョブ FEKSETUP は、リストされているすべてのメンバーを別のデータ・セットおよびディレクトリー (デフォルトでは FEK.#CUST.* および /etc/rdz/*) にコピーします。

表 48. バージョン 7.5 のカスタマイズ

メンバー/ファイル	デフォルト・ロケーション	目的	マイグレーションの注
FEKSETUP	FEK.SFEKSAMP [FEK.#CUST.JCL]	データ・セットおよびディレクトリーを作成し、カスタマイズ可能ファイルのデータを取り込むための JCL	新規。カスタマイズが必要
JMON	FEK.SFEKSAMP (FEJJJCL) [FEK.#CUST.PROCLIB]	JES ジョブ・モニター用の JCL	STEPLIB が SFEKAUTH に変更された
RSED	FEK.SFEKSAMP (FEKRSED) [FEK.#CUST.PROCLIB]	RSE デーモンの JCL	新規。カスタマイズが必要
ELAXF*	FEK.SFEKSAMP [FEK.#CUST.PROCLIB]	リモート・プロジェクト・ビルドなどのための JCL	ELAXFTSO、ELAXFCPI、および ELAXFPP1 は新規
FEKRACF	FEK.SFEKSAMP [FEK.#CUST.JCL]	セキュリティ定義の JCL	新規、必須
FEJJCNFG	FEK.SFEKSAMP [FEK.#CUST.PARMLIB]	JES ジョブ・モニター構成ファイル	<ul style="list-style-type: none"> • 一部のディレクティブがオプションになった • オプションの新規ディレクティブが追加された
FEJTSO	FEK.SFEKSAMP [FEK.#CUST.CNTL]	TSO 実行依頼用の JCL	ジョブ名を変数にできるようになった

表 48. パージョン 7.5 のカスタマイズ (続き)

メンバー/ファイル	デフォルト・ロケーション	目的	マイグレーションの注
CRAISPRX	FEK.SFEKSAMP [FEK.#CUST.CNTL]	TSO/ISPF クライアント・ゲートウェイを使用した CARMA 用のサンプル DD 割り振り exec	新規、カスタマイズはオプション
CRAXJCL	FEK.SFEKSAMP [FEK.#CUST.ASM]	IRXJCL 置換用のサンプル・ソース・コード	新規、カスタマイズはオプション
CRA#CIRX	FEK.SFEKSAMP [FEK.#CUST.JCL]	CRAXJCL をコンパイルするための JCL	新規、カスタマイズはオプション
ADNSMSGH	FEK.SFEKSAMP [FEK.#CUST.COBOLE]	パイプライン・メッセージ・ハンドラー用のサンプル・ソース・コード	古いコピーは、このコピーに置き換える必要がある (カスタマイズの再実行が必要)
ADNPCCSD	FEK.SFEKSAMP [FEK.#CUST.JCL]	CRD サーバーを主 CICS 領域に対して定義するための JCL	古いコピーは、このコピーに置き換える必要がある (カスタマイズの再実行が必要)
ADNJSPAU	FEK.SFEKSAMP [FEK.#CUST.JCL]	CRD のデフォルトを更新するための JCL	新規、カスタマイズはオプション
ADNMFEST	FEK.SFEKSAMP [FEK.#CUST.JCL]	マニフェスト・リポジトリを作成し、定義するための JCL	新規、カスタマイズはオプション
rsed.envvars	/usr/lpp/rdz/samples/ [/etc/rdz/]	RSE 環境変数	古いコピーは、このコピーに置き換える必要がある (カスタマイズの再実行が必要)
ISPF.conf	/usr/lpp/rdz/samples/ [/etc/rdz/]	TSO/ISPF クライアント・ゲートウェイ構成ファイル	v7.1 で SCLMDT に添付されていた ISPF.conf と同じもの
CRASRV.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	CARMA 構成ファイル	<ul style="list-style-type: none"> ・ 始動スクリプトのロケーションと名前が変更された ・ オプションの新規ディレクティブが追加された
crastart.conf	/usr/lpp/rdz/samples/ [/etc/rdz/]	CRASTART を使用するための CARMA 構成ファイル	新規、カスタマイズはオプション

表 48. バージョン 7.5 のカスタマイズ (続き)

メンバー/ファイル	デフォルト・ロケーション	目的	マイグレーションの注
FMEXT.properties	/usr/lpp/rdz/samples/ [/etc/rdz/]	File Manager Integration 構成ファイル	<ul style="list-style-type: none"> ・ 始動スクリプトのロケーションと名前が変更された ・ オプションの新規ディレクティブが追加された
uchars.settings	/usr/lpp/rdz/samples/ [/etc/rdz/]	編集不可能文字構成ファイル	新規、カスタマイズはオプション

バージョン 7.0 からバージョン 7.1 へのマイグレーション

IBM Rational Developer for System z、FMID HHOP710

- ・ SMP/E による MVS および z/OS UNIX コンポーネントのデフォルトのインストール・ロケーションは、変更されておらず、したがって FEK.* および /usr/lpp/wd4z/* のままです。
- ・ 追加: セットアップの選択 - APPC トランザクションまたは SCLM Developer Toolkit を介した TSO/ISPF コマンド
- ・ 変更: APPC トランザクションは新しい ISPF フィーチャーを活用する
- ・ 追加: 以下のカスタマイズ可能メンバーは新規。
 - samplib ELAXFADT
 - samplib ADNCMSGH
 - /usr/lpp/wd4z/rse/lib/FMEXT.properties
- ・ 変更: 以下のメンバーが移動した。
 - SFEKDLL(FEJBDTRX) -> SFEKLOAD(FEJBDTRX)
- ・ 変更: 以下のカスタマイズ可能メンバーが変更された。
 - samplib FEKFAPPCC
 - /usr/lpp/wd4z/rse/lib/rsed.envvars
 - /usr/lpp/wd4z/rse/lib/setup.env.zseries
 - /usr/lpp/wd4z/rse/lib/server.zseries

IBM 共通アクセス・リポジトリ・マネージャー (CARMA)、FMID HCMA710

- ・ SMP/E による MVS コンポーネントのデフォルトのインストール・ロケーションは、変更されておらず、したがって CRA.* のままです。
- ・ 変更: ログが CARMALOG DD ステートメントへ書き込まれる
- ・ 変更: CARMA メッセージ VSAM (CRAMSG) および構成 VSAM (CRADEF) が更新された

- **追加:** 以下のカスタマイズ可能メンバーは新規。
 - samplib CRA#ECOB
 - samplib CRA#EPDS
 - samplib CRA#ERAM
 - samplib CRA#ESLM
- **名前変更:** 以下のカスタマイズ可能メンバーは名前が変更された。
 - samplib CRAREPR -> CRA\$VDEF
 - samplib CRAMREPR -> CRA\$VMSG
 - samplib CRASREPR -> CRA\$VSTR
 - samplib CRASALX -> CRA#ASLM
 - samplib CRACOBJ1 -> CRA#CCB1
 - samplib CRACOBJ2 -> CRA#CCB2
 - samplib CRACLICM -> CRA#CCLT
 - samplib CRARAMCS -> CRA#CPDS
 - samplib CRARAMCM -> CRA#CRAM
 - samplib CRATREPR -> CRA#VPDS
 - samplib CRALREPR -> CRA#VSLM
 - samplib CRACLIRN -> CRA#XCLT
- **変更:** 以下のカスタマイズ可能メンバーが変更された。
 - clist CRASUBMT

構成可能なファイル

表 23 は、バージョン 7.1 でカスタマイズされるファイルの概要を示しています。CARMA および Developer for System z のサンプル・ライブラリー CRA.SCRASAMP、FEK.SFEKSAMP、および /usr/lpp/wd4z/rse/lib/ は、ここにリストしたものより多くのカスタマイズ可能なメンバーが付属する場合があることに注意してください (サンプルの CARMA ソース・コードおよびそれらをコンパイルするジョブなど)。

表 49. バージョン 7.1 のカスタマイズ

メンバー/ファイル	デフォルト・ロケーション	目的	マイグレーションの注
ELAXF*	FEK.SFEKSAMP	リモート・プロジェクト・ビルドおよびその他のジョブ用の JCL	ELAXFADT は新規
CRA\$VMSG	CRA.SCRASAMP	CARMA メッセージ VSAM を作成するための JCL	<ul style="list-style-type: none"> • 名前変更。旧 CRAMREPR • このジョブで作成される VSAM が更新されました。

表 49. バージョン 7.1 のカスタマイズ (続き)

メンバー/ファイル	デフォルト・ロケーション	目的	マイグレーションの注
CRA\$VDEF	CRA.SCRASAMP	CARMA 構成 VSAM を作成するための JCL	<ul style="list-style-type: none"> 名前変更。旧 CRAREPR このジョブで作成される VSAM が更新されました。
CRA\$VSTR	CRA.SCRASAMP	CARMA カスタム情報 VSAM を作成するための JCL	名前変更。旧 CRASREPR
CRASUBMT	CRA.SCRASAMP	CARMA パッチ始動 CLIST	DD CARMALOG を追加
CRA#VSLM	CRA.SCRASAMP	SCLM RAM のメッセージ VSAM を作成するための JCL	名前変更。旧 CRALREPR
CRA#ASLM	CRA.SCRASAMP	SCLM RAM のデータ・セットを作成するための JCL	名前変更。旧 CRASALX
CRA#VPDS	CRA.SCRASAMP	PDS RAM のメッセージ VSAM を作成するための JCL	名前変更。旧 CRATREPR
CRA#CRAM	CRA.SCRASAMP	スケルトン RAM をコンパイルするための JCL	名前変更。旧 CRARAMCM
FEKAPPCC	FEK.SFEKSAMP	APPC トランザクションを作成するための JCL	ISPF NEST サポートを活用
rased.envvars	/usr/lpp/wd4z/rse/lib/ [/etc/wd4z/]	RSE 環境変数	古いコピーは、このコピーに置き換える必要がある (カスタマイズの再実行が必要)
FMIEXT.properties	/usr/lpp/wd4z/rse/lib/ [/etc/wd4z/]	File Manager Integration 構成ファイル	新規。使用する場合はカスタマイズが必要

付録 A. SSL および X.509 認証のセットアップ

この付録は、Secure Socket Layer (SSL) のセットアップ時、または既存のセットアップの検査時や変更時に起きる可能性があるいくつかの一般的な問題について、支援を提供するためのものです。また、この付録には、X.509 証明書で自分自身を認証するユーザーをサポートするためのサンプルのセットアップも記載されています。

セキュアな通信は、正しい通信パートナーと通信できるようにし、他の人間がデータを傍受して読み取ることが困難な方法で情報を伝送することを意味します。SSL は、TCP/IP ネットワーク内でその機能を提供します。SSL では、デジタル証明書を使用して自分自身を識別し、公開鍵プロトコルを使用して通信を暗号化します。SSL で使用されるデジタル証明書および公開鍵プロトコルの詳細については、「*Security Server RACF セキュリティー管理者のガイド*」(SA88-8613) を参照してください。

Developer for System z 用に SSL 通信をセットアップするために必要なアクションは、厳密な必要性、使用される RSE 通信方式、およびサイトで既に使用可能なものに応じてサイトごとに異なります。

この付録では、現行の RSE 定義のクローンを作成して、SSL を使用する 2 番目の RSE デーモン接続を使用できるようにします。また、RSE 接続のさまざまな部分で使用される独自のセキュリティ証明書も作成します。

- 316 ページの『秘密鍵と証明書を保管する場所の決定』
- 317 ページの『RACF による鍵リングの作成』
- 319 ページの『既存の RSE セットアップのクローン作成』
- 319 ページの『共存を可能にするための rsed.envvars の更新』
- 320 ページの『SSL を有効にするための ssl.properties の更新』
- 320 ページの『新しい RSE デーモンの作成による SSL のアクティブ化』
- 321 ページの『接続のテスト』
- 324 ページの『(オプション) X.509 クライアント認証サポートの追加』
- 325 ページの『(オプション) gskkyman による鍵データベースの作成』
- 327 ページの『(オプション) keytool による鍵ストアの作成』

この付録では、次のような一律の命名規則が使用されています。

- 証明書: rdzrse
- 鍵および証明書ストレージ: rdzssl.*
- パスワード: rsessl
- デーモン・ユーザー ID: stcrse

以下に述べるタスクの一部では、ユーザーが z/OS UNIX 内でアクティブであることを想定しています。そのためには、TSO コマンド **OMVS** を発行します。TSO に戻るには、**exit** コマンドを使用します。

秘密鍵と証明書を保管する場所の決定

SSL で使用される ID 証明書と暗号化/暗号化解除鍵は、鍵ファイルに保管されます。この鍵ファイルには、アプリケーション・タイプに応じて、さまざまな実装が存在します。

しかし、すべての実装が同じ原則に従います。あるコマンドが、鍵ペア（公開鍵とそれに関連付けられた秘密鍵）を生成します。その後、コマンドは公開鍵を X.509 自己署名証明書の中に包み込み、この証明書は単一要素の証明書チェーンとして保管されます。この証明書チェーンと秘密鍵が（別名によって識別される）1 つの項目として、鍵ファイルに保管されます。

RSE デーモンは System SSL アプリケーションであり、鍵データベース・ファイルを使用します。この鍵データベースは、gskkyman によって作成された物理ファイルとするか、SAF 準拠のセキュリティ・ソフトウェア（例えば RACF）によって管理される鍵リングとすることができます。RSE サーバー（これはデーモンによって始動されます）は Java SSL アプリケーションであり、keytool によって作成される鍵ストア・ファイルを使用するか、セキュリティ・ソフトウェアによって管理される鍵リングを使用します。

表 50. SSL 証明書の保管メカニズム

証明書ストレージ	作成者および管理者	RSE デーモン	RSE サーバー
鍵リング	SAF 準拠のセキュリティ製品	サポート	サポート
鍵データベース	z/OS UNIX の gskkyman	サポート	/
鍵ストア	Java の keytool	/	サポート

SSL を通じて接続するためには、鍵ストアと鍵データベースの両方が、z/OS UNIX ファイルとして、または SAF 準拠の鍵リングとして必要です。

- 鍵ストア (RACF または keytool)
- 鍵データベース (RACF または gskkyman)

注:

- SAF 準拠の鍵リングは、証明書の管理に推奨される方式です。
- RSE デーモンと RSE サーバーが同じ証明書管理方式を使用する場合は、共用証明書を使用できます。
- RSE デーモンは、プログラム制御で実行する必要があります。内部で System SSL を使用することは、SYS1.SIEALNKE が、ご使用のセキュリティ・ソフトウェアによってプログラム制御されている必要があることを意味します。
- System SSL アプリケーション（デーモン接続）を実行するためには、SYS1.SIEALNKE が LINKLIST または STEPLIB の中に存在する必要があります。STEPLIB 方式を使用したい場合は、次のステートメントを rsed.envvars の末尾に追加します。

```
STEPLIB=$STEPLIB:SYS1.SIEALNKE
```

ただし、以下の点に注意してください。

- STEPLIB を z/OS UNIX で使用すると、パフォーマンスに悪い影響が出ます。
- 1 つの STEPLIB ライブラリーに APF 許可がある場合、すべての STEPLIB ライブラリーに APF 許可があることが必要です。ライブラリーは、STEPLIB 内で許可のないライブラリーと混用した場合、APF 許可を失います。
- System SSL は、Integrated Cryptographic Service Facility (ICSF) を使用します (使用可能な場合)。ICSF は、System SSL ソフトウェア・アルゴリズムの代わりに使用されるハードウェア暗号化サポートを提供します。詳細については、「System SSL プログラミング」(SD88-6252) を参照してください。

RACF およびデジタル証明書については、「Security Server RACF セキュリティー管理者のガイド」(SA88-8613) を参照してください。gskkyman の資料は、「System SSL プログラミング」(SD88-6252) で、keytool の資料は、<http://java.sun.com/j2se/1.5.0/docs/tooldocs/solaris/keytool.html> で入手できます。

RACF による鍵リングの作成

RSE デーモン鍵データベースの作成に gskkyman を使用し、RSE サーバー鍵ストアの作成に keytool を使用する場合は、このステップを実行しないでください。

RACDCERT コマンドは、秘密鍵と証明書を RACF にインストールし、保守します。RACF は、複数の秘密鍵と証明書を 1 つのグループとして管理できます。それらのグループは、鍵リングと呼ばれます。

RACDCERT コマンドの詳細については、「Security Server RACF コマンド言語解説書」(SA88-8617) を参照してください。

```

RDEFINE FACILITY IRR.DIGTCERT.LIST UACC(NONE)
RDEFINE FACILITY IRR.DIGTCERT.LISTRING UACC(NONE)
PERMIT IRR.DIGTCERT.LIST CLASS(FACILITY) ACCESS(READ) ID(stcrse)
PERMIT IRR.DIGTCERT.LISTRING CLASS(FACILITY) ACCESS(READ) ID(stcrse)
SETOPTS RACLIST(FACILITY) REFRESH

RACDCERT ID(stcrse) GENCERT SUBJECTSDN(CN('rdz rse ssl') +
OU('rdz') O('IBM') L('Raleigh') SP('NC') C('US')) +
NOTAFTER(DATE(2017-05-21)) WITHLABEL('rdzrse') KEYUSAGE(HANDSHAKE)

RACDCERT ID(stcrse) ADDRING(rdzssl.racf)
RACDCERT ID(stcrse) CONNECT(LABEL('rdzrse') RING(rdzssl.racf) +
DEFAULT USAGE(PERSONAL))

```

上記のサンプルは始めに、必要なプロファイルを作成し、ユーザー ID STCRSE に、そのユーザー ID が所有する証明書と鍵リングへのアクセスを許可します。使用するユーザー ID は、SSL RSE デーモンを実行するために使用したユーザー ID に一致する必要があります。次のステップは、新しい自己署名証明書を rdzrse というラベルで作成することです。パスワードは必要ありません。この証明書は、その後、新規に作成された鍵リング (rdzssl.racf) に追加されます。証明書の場合と同様に、鍵リングにパスワードは必要ありません。

結果は、以下の list オプションを使用して検査できます。

```

RACDCERT ID(stcrse) LIST
Digital certificate information for user STCRSE:

```

```

Label: rdzrse
Certificate ID: 2QjW10Xi0sXZ1aaEqZmihUBA
Status: TRUST
Start Date: 2007/05/24 00:00:00
End Date: 2017/05/21 23:59:59
Serial Number:
>00<
Issuer's Name:
>CN=rdz rse ssl.OU=rdz.O=IBM.L=Raleigh.SP=NC.C=US<
Subject's Name:
>CN=rdz rse ssl.OU=rdz.O=IBM.L=Raleigh.SP=NC.C=US<
Private Key Type: Non-ICSF
Private Key Size: 1024
Ring Associations:
Ring Owner: STCRSE
Ring:
>rdzssl.racf<

```

(オプション) 署名付き証明書の使用

証明書には、自己署名付きの証明書と認証局 (CA) によって署名された証明書があります。CA によって署名された証明書は、その証明書の所有者が本人に間違いのないことを CA が保証することを意味します。署名プロセスでは、CA 資格情報 (証明書でもある) がユーザーの証明書に追加され、複数要素の証明書チェーンが作成されます。

CA によって署名された証明書を使用した場合は、Developer for System z クライアントによる信頼性検証のための質問を回避できます (クライアントがすでにその CA を信頼している場合)。

CA 署名付き証明書を作成および使用するには、以下のステップを実行します。

1. 自己署名証明書を作成します。

```
RACDCERT ID(stcrse) GENCERT WITHLABEL('rdzrse') . . .
```

2. この証明書についての署名要求を作成します。

```
RACDCERT ID(stcrse) GENREQ (LABEL('rdzrse')) DSN(dsn)
```

3. 署名要求を、選択した CA に送信します。

4. CA 資格情報 (証明書でもある) が既知であるかどうかを検査します。

```
RACDCERT CERTAUTH LIST
```

5. CA 証明書に、信頼できるものとしてのマークを付けます。

```
RACDCERT CERTAUTH ALTER(LABEL('CA cert')) TRUST
```

または、CA 証明書をデータベースに追加します。

```
RACDCERT CERTAUTH ADD(dsn) WITHLABEL('CA cert') TRUST
```

6. 署名付き証明書をデータベースに追加します。これにより、自己署名証明書が置き換えられます。

```
RACDCERT ID(stcrse) ADD(dsn) WITHLABEL('rdzrse') TRUST
```

注: 自己署名証明書を、置き換える前に削除しないでください。削除すると、証明書に付随する秘密鍵が失われ、証明書が役に立たなくなります。

7. 鍵リングを作成します。

```
RACDCERT ID(stcrse) ADDRING(rdzssl.racf)
```

8. 署名付き証明書を鍵リングに追加します。


```
RACDCERT ID(stcrse) CONNECT(ID(stcrse) LABEL('rdzrse')  
RING(rdzssl.racf))
```

9. CA 証明書を鍵リングに追加します。

```
RACDCERT ID(stcrse) CONNECT(CERTAUTH LABEL('CA cert')  
RING(rdzssl.racf))
```

既存の RSE セットアップのクローン作成

このステップでは、SSL セットアップを既存のセットアップと並行して稼働できるように、RSE 構成ファイルの新しいインスタンスを作成します。以下のサンプル・コマンドでは、構成ファイルが `/etc/rdz/` に入っていることを想定しています。これは、18 ページの『カスタマイズのセットアップ』で使用されるデフォルトのロケーションです。

```
$ cd /etc/rdz  
$ mkdir ssl  
$ cp rsed.envvars ssl  
$ cp ssl.properties ssl  
$ ls ssl  
rsed.envvars    ssl.properties
```

上記の z/OS UNIX コマンドは `ssl` というサブディレクトリを作成し、そこに変更が必要な構成ファイルを取り込みます。その他の構成ファイル、インストール・ディレクトリ、および MVS コンポーネントは、SSL 固有ではないので、共用できます。

既存の大部分の構成ファイルを再利用することにより、SSL のセットアップに実際に必要な変更集中でき、RSE セットアップ全体を再度実行しなくても済みます。(例えば、`ISPF.conf` の新しいロケーションを定義せずに済みます。)

共存を可能にするための rsed.envvars の更新

これまでのところ、定義は現行のセットアップの正確なコピーであり、これは、新しい RSE デーモンのログが現行のサーバー・ログ・ファイルと重なることを意味します。また、RSE は、`ssl` ディレクトリにコピーされなかった構成ファイルがどこにあるかも知っている必要があります。どちらの問題も、`rsed.envvars` に小さな変更を加えることによって対処できます。

```
$ oedit /etc/rdz/ssl/rsed.envvars  
-> uncomment and change: -Ddaemon.log=/var/rdz/logs/ssl  
-> add at the END:  
# -- NEEDED TO FIND THE REMAINING CONFIGURATION FILES  
CFG_BASE=/etc/rdz  
CLASSPATH=.:$CFG_BASE:$CLASSPATH  
# --
```

上記の変更では、新しいログ・ロケーションが定義されます (そのログ・ロケーションは、存在しない場合、RSE デーモンによって作成されます)。また、これらの変更では、SSL RSE プロセスが最初に現行ディレクトリ (`/etc/rdz/ssl`) で構成ファイルを検索し、その後に元のディレクトリ (`/etc/rdz`) を検索するように `CLASSPATH` も更新されます。

SSL を有効にするための ssl.properties の更新

ssl.properties を更新することにより、SSL 暗号化通信を使用して開始するように RSE に指示します。

```
$ oedit /etc/rdz/ssl/ssl.properties
-> change: enable_ssl=true
-> uncomment and change: daemon_keydb_file=rdzssl.racf
-> uncomment and change: daemon_key_label=rdzrse
-> uncomment and change: server_keystore_file=rdzssl.racf
-> uncomment and change: server_keystore_label=rdzrse
-> uncomment and change: server_keystore_type=JCERACFKS
```

上記の変更は SSL を使用可能に設定し、RSE デーモンおよび RSE サーバーに、(共用の) 証明書が鍵リング rdzssl.racf 内のラベル rdzrse の下に保管されることを通知します。JCERACFKS キーワードは、SAF 準拠の鍵リングが鍵ストアとして使用されることを RSE サーバーに通知します。

新しい RSE デーモンの作成による SSL のアクティブ化

前に述べたように、ここでは SSL を使用する第 2 の接続を作成します。これは、新しい RSE デーモンを作成することを意味します。RSE デーモンは、開始タスクまたはユーザー・ジョブとすることができます。ここでは、初期 (テスト) セットアップにユーザー・ジョブ方式を使用します。以下の説明では、サンプル JCL が FEK.#CUST.PROCLIB(RSED) に入っていることを想定しています。これは、18 ページの『カスタマイズのセットアップ』で使用されるデフォルトのロケーションです。

1. 新しいメンバー FEK.#CUST.PROCLIB(RSESSL) を作成し、サンプル JCL FEK.#CUST.PROCLIB(RSED) の中にコピーします。
2. 先頭にジョブ・カードを、末尾に exec ステートメントをそれぞれ追加して、RSESSL をカスタマイズします。また、以下のコード・サンプルに示すように、SSL に関連する構成ファイル (/etc/rdz/ssl) の新しいポート番号 (4047) とロケーションも指定します。ここでは、ユーザー ID STCRSE を使用する点に注意してください。このユーザー ID には、前のステップで、証明書と鍵リングに対する正しいアクセス権限が与えられているためです。

```
//RSESSL JOB CLASS=A,MSGCLASS=A,MSGLEVEL=(1,1),USER=STCRSE
//*
/* RSE DAEMON - SSL
/*
//RSED PROC IVP='', * 'IVP' to do an IVP test
// PORT=4047,
// HOME='/usr/lpp/rdz',
// CNFG='/etc/rdz/ssl'
/*
//RSE EXEC PGM=BPXBATSL,REGION=0M,TIME=NOLIMIT,
// PARM='PGM &HOME./bin/rsed.sh &IVP &PORT &CNFG'
//STDOUT DD SYSOUT=*
//STDERR DD SYSOUT=*
//PEND
/*
//RSED EXEC RSED
/*
```

図 61. RSESSL - SSL 用の RSE デーモン・ユーザー・ジョブ

注: RSEDSSL ジョブに割り当てられるユーザー ID は、元のRSE デーモンと同じ権限を持っている必要があります。FACILITY プロファイル BPX.SERVER と PTKTDATA プロファイル IRRPTAUTH.FEKAPPL.* が、ここでの重要な要素です。

接続のテスト

SSL ホスト構成が完了し、前に作成したジョブ FEK.#CUST.PROCLIB(RSEDSSL) を実行依頼することにより、SSL 用の RSE デーモンを始動できます。

この時点で、Developer for System z クライアントと接続することにより、新しいセットアップをテストできます。SSL で使用するために新しい構成を（既存の構成のクローン作成によって）作成してあるので、RSE デーモン用のポート 4047 を使用して、クライアント上に新しい接続を定義する必要があります。

接続と同時に、ホストとクライアントはセキュア・パスをセットアップするために何らかのハンドシェークを開始します。このハンドシェークの一環として、証明書の交換があります。Developer for System z クライアントは、ホスト証明書またはそれに署名した CA を認識できない場合、ユーザーにその証明書が信頼できるかどうかを尋ねるプロンプトを出します。

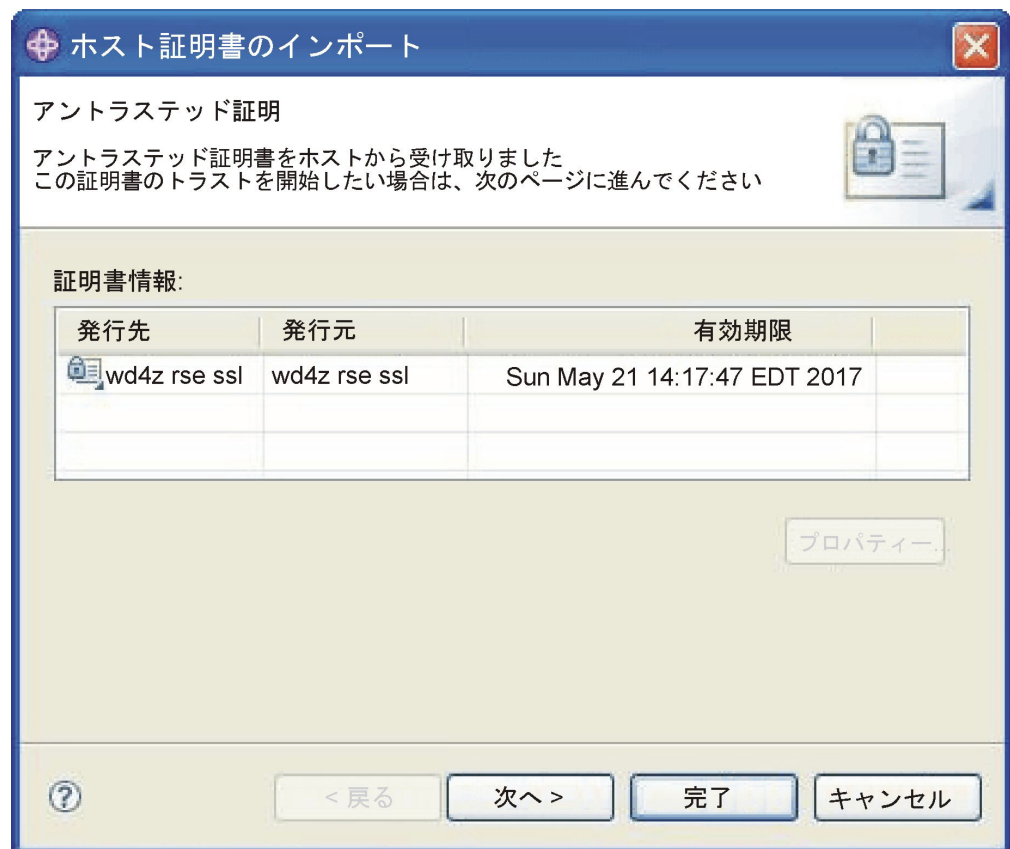


図 62. 「ホスト証明書のインポート」 ダイアログ

「完了」 ボタンをクリックすると、この証明書を信頼できるものとして受け入れることができ、その後、接続の初期化が続行されます。

注: RSE デーモンと RSE サーバーは 2 つの異なる証明書ロケーションを使用する場合があり、その結果、2 つの異なる証明書が使用され、したがって 2 つの確認が行われる場合があります。

証明書がクライアントに既知となった後、このダイアログが再び表示されることはありません。信頼できる証明書のリストは、「ウィンドウ」>「設定...」>「リモート・システム」>「SSL」を選択することによって管理できます。その場合、次のダイアログが表示されます。

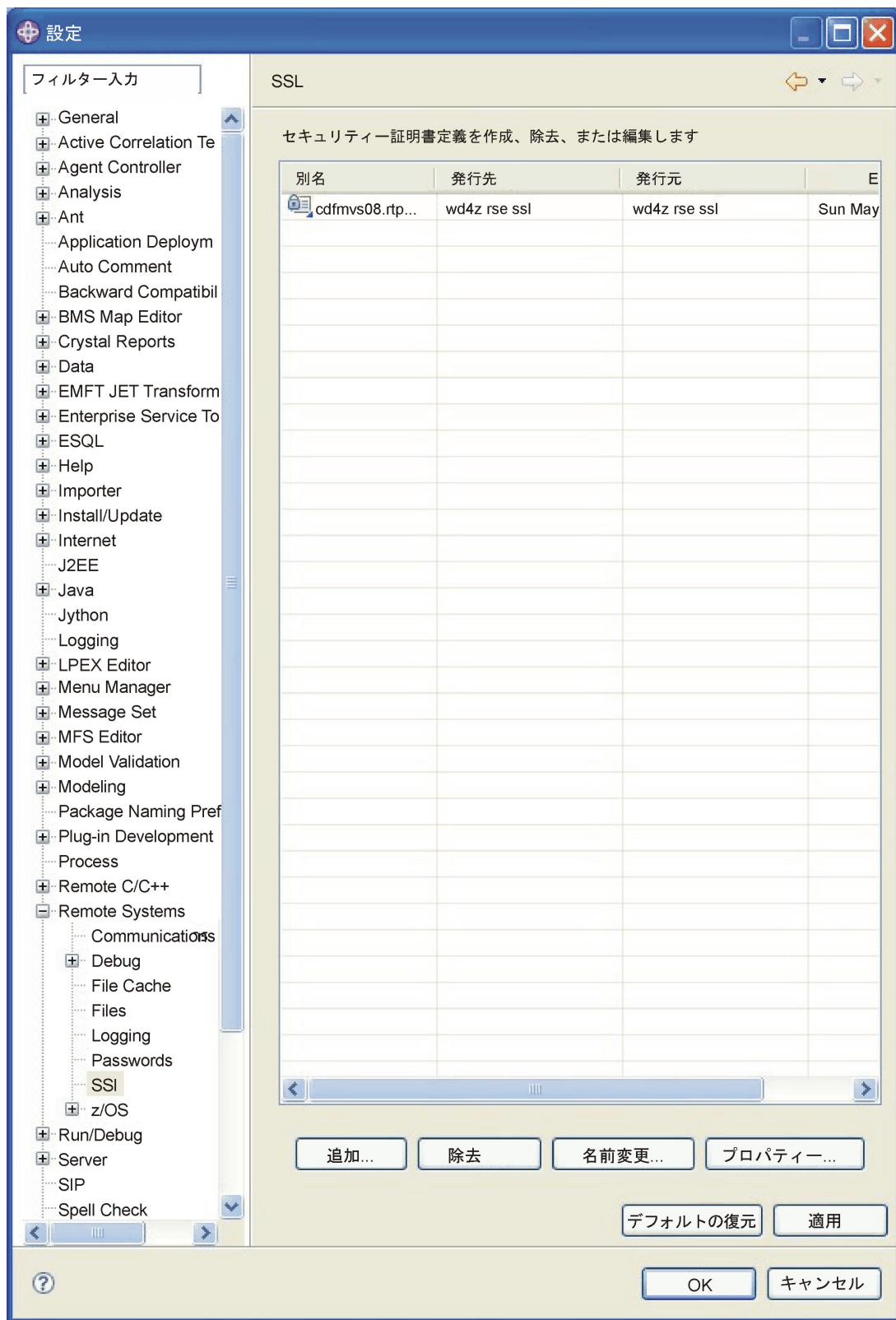


図 63. 「設定」 ダイアログ - 「SSL」

SSL 通信が失敗した場合、クライアントはエラー・メッセージを返します。詳細な情報は、148 ページの『RSE デーモンおよびスレッド・プールのロギング』、およ

び 148 ページの『RSE ユーザー・ロギング』の説明のように、さまざまなサーバーおよびユーザー・ログ・ファイルから入手できます。

(オプション) X.509 クライアント認証サポートの追加

RSE デーモンは、X.509 証明書で自分自身を認証するユーザーをサポートしています。この機能の場合、SSL 暗号化通信の使用は前提条件です。この機能は、SSL で使用される証明書でのホスト認証を拡張したものです。

ユーザーの証明書認証を行うには、182 ページの『X.509 証明書を使用したクライアント認証』の説明にあるように、複数の方法があります。以下の手順は、セキュリティー・ソフトウェアで HostIdMappings 証明書拡張を使用して証明書を認証する方式をサポートするために必要なセットアップを文書化したものです。

1. クライアント証明書への署名に使用された認証局 (CA) を識別する証明書を、高度に信頼できる CA 証明書に変更します。証明書の妥当性検査には TRUST 状況で十分ですが、ログオン・プロセスの証明書認証の部分で HIGHTRUST が使用されるので、HIGHTRUST に変更します。

```
RACDCERT CERTAUTH ALTER(LABEL('HighTrust CA')) HIGHTRUST
```

2. CA 証明書を鍵リング rdzssl.racf に追加します。これにより、CA 証明書をクライアント証明書の妥当性検査に使用できるようになります。

```
RACDCERT ID(stcrse) CONNECT(CERTAUTH LABEL('HighTrust CA') +  
RING(rdzssl.racf))
```

これで、CA 証明書のためのセキュリティー・ソフトウェアのセットアップは完了しました。

3. クライアント証明書の HostIdMappings 拡張で定義されているホスト名 CDFMVS08.RALEIGH.IBM.COM 用に、SERVAUTH クラス内にリソースを定義します (フォーマット IRR.HOST.hostname)。

```
RDEFINE SERVAUTH IRR.HOST.CDFMVS08.RALEIGH.IBM.COM UACC(NONE)
```

4. RSE 開始タスク・ユーザー ID STCRSE に、このリソースに対する READ アクセス権限を付与します。

```
PERMIT IRR.HOST.CDFMVS08.RALEIGH.IBM.COM CLASS(SERVAUTH) +  
ACCESS(READ) ID(stcrse)
```

5. SERVAUTH クラスに対する変更をアクティブにします。SERVAUTH クラスがまだアクティブでない場合は、最初のコマンドを使用します。アクティブなセットアップをリフレッシュするには、2 番目のコマンドを使用します。

```
SETOPTS CLASSACT(SERVAUTH) RACLIST(SERVAUTH)  
or  
SETOPTS RACLIST(SERVAUTH) REFRESH
```

これで、HostIdMappings 拡張のためのセキュリティー・ソフトウェアのセットアップは完了しました。

6. RSE 開始タスクを再始動して、X.509 証明書を使用したクライアント・ログオンの受け入れを開始します。

(オプション) gskkyman による鍵データベースの作成

RSE デモンの鍵データベースに SAF 準拠の鍵リングを使用する場合は、このステップを実行しないでください。

gskkyman は z/OS UNIX シェル・ベースのメニュー方式プログラムであり、秘密鍵、証明書要求、および証明書が入っている z/OS UNIX ファイルを作成し、それにデータを取り込み、管理します。この z/OS UNIX ファイルは鍵データベースと呼ばれます。

注: gskkyman 用の環境をセットアップするために、以下のステートメントが必要になる場合があります。これについての詳細は、「*System SSL プログラミング*」(SD88-6252) を参照してください。

```
PATH=$PATH:/usr/lpp/gskssl/bin
export NLSPATH=/usr/lpp/gskssl/lib/nls/msg/En_US.IBM-1047/%N:$NLSPATH
export STEPLIB=$STEPLIB:SYS1.SIEALNKE
```

```
$ cd /etc/rdz/ssl
$ gskkyman          Database Menu
```

1 - Create new database

```
Enter option number: 1
Enter key database name (press ENTER to return to menu): rdzssl.kdb
Enter database password (press ENTER to return to menu): rsessl
Re-enter database password: rsessl
Enter password expiration in days (press ENTER for no expiration):
Enter database record length (press ENTER to use 2500):
```

Key database /etc/rdz/ssl/rdzssl.kdb created.

Press ENTER to continue.

Key Management Menu

6 - Create a self-signed certificate

Enter option number (press ENTER to return to previous menu): 6

Certificate Type

5 - User or server certificate with 1024-bit RSA key

```
Select certificate type (press ENTER to return to menu): 5
Enter label (press ENTER to return to menu): rdzrse
Enter subject name for certificate
  Common name (required): rdz rse ssl
  Organizational unit (optional): rdz
  Organization (required): IBM
  City/Locality (optional): Raleigh
  State/Province (optional): NC
  Country/Region (2 characters - required): US
Enter number of days certificate will be valid (default 365): 3650

Enter 1 to specify subject alternate names or 0 to continue: 0

Please wait .....

Certificate created.

Press ENTER to continue.
```

Key Management Menu

0 - Exit program

Enter option number (press ENTER to return to previous menu): 0

\$ ls -l rdzssl.*

total 152

-rw----- 1 IBMUSER SYS1 35080 May 24 14:24 rdzssl.kdb

-rw----- 1 IBMUSER SYS1 80 May 24 14:24 rdzssl.rdb

\$ chmod 644 rdzssl.*

\$ ls -l rdzssl.*

-rw-r--r-- 1 IBMUSER SYS1 35080 May 24 14:24 rdzssl.kdb

-rw-r--r-- 1 IBMUSER SYS1 80 May 24 14:24 rdzssl.rdb

上記のサンプルは始めに、rdzssl.kdb という鍵データベースを、パスワード rssql を使用して作成します。データベースの作成後、約 10 年間 (うるう日はカウントしません) 有効な新しい自己署名証明書を作成することにより、データベースにデータが取り込まれます。証明書は、rdzrse というラベルの下に、鍵データベースに使用されたのと同じパスワード (rssql) を使用して保管されます (これは RSE の必要条件です)。

gskkyman は、鍵データベースに (非常にセキュアな) 600 許可ビット・マスク (所有者だけがアクセスできる) を割り振ります。デーモンが鍵データベースの作成者と同じユーザー ID を使用している場合を除き、アクセス権限の設定をもっと制限の少ないものにする必要があります。chmod コマンドには、644 (所有者は読み取り/書き込み権限を持ち、全員が読み取り権限を持つ) のマスクを使用できます。

結果を検査するには、以下のように、「**Manage keys and certificates**」サブメニューで「**Show certificate information**」オプションを選択します。

\$ gskkyman

Database Menu

2 - Open database

Enter option number: 2

Enter key database name (press ENTER to return to menu): rdzssl.kdb

Enter database password (press ENTER to return to menu): rssql

Key Management Menu

1 - Manage keys and certificates

Enter option number (press ENTER to return to previous menu): 1

Key and Certificate List

1 - rdzrse

Enter label number (ENTER to return to selection menu, p for previous list): 1

Key and Certificate Menu

1 - Show certificate information

Enter option number (press ENTER to return to previous menu): 1

Certificate Information

Label: rdzrse

Record ID: 14

Issuer Record ID: 14

```
Trusted: Yes
Version: 3
Serial number: 45356379000ac997
Issuer name: rdz rse ssl
rdz
IBM
Raleigh
NC
US
Subject name: rdz rse ssl
rdz
IBM
Raleigh
NC
US
Effective date: 2007/05/24
Expiration date: 2017/05/21
Public key algorithm: rsaEncryption
Public key size: 1024
Signature algorithm: sha1WithRsaEncryption
Issuer unique ID: None
Subject unique ID: None
Number of extensions: 3
```

Enter 1 to display extensions, 0 to return to menu: 0

Key and Certificate Menu

0 - Exit program

Enter option number (press ENTER to return to previous menu): 0

次の ssl.properties のサンプルは、daemon_* ディレクティブが、前に示した SAF 鍵リングのサンプルと異なることを示しています。

```
$ oedit /etc/rdz/ssl/ssl.properties
-> change: enable_ssl=true
-> uncomment and change: daemon_keydb_file=rdzssl.kdb
-> uncomment and change: daemon_keydb_password=rsessl
-> uncomment and change: daemon_key_label=rdzrse
-> uncomment and change: server_keystore_file=rdzssl.racf
-> uncomment and change: server_keystore_label=rdzrse
-> uncomment and change: server_keystore_type=JCERACFKS
```

上記の変更は SSL を使用可能に設定し、RSE デーモンに、パスワード rsessl を持つ鍵データベース rdzssl.kdb 内のラベル rdzrse の下に証明書が保管されることを通知します。RSE サーバーは、依然として SAF 準拠の鍵リングを使用しています。

(オプション) keytool による鍵ストアの作成

RSE サーバーの鍵ストアに SAF 準拠の鍵リングを使用する場合は、このステップを実行しないでください。

「keytool -genkey」は、秘密鍵ペアとそれに適合する自己署名証明書を生成します。それらは、(別名によって識別される) 1 つの項目として (新しい) 鍵ストア・ファイルに保管されます。

注: コマンド検索ディレクトリーに Java が含まれている必要があります。keytool を実行するには、以下のステートメントが必要になる場合があります。ここ

で、/usr/lpp/java/J5.0 は Java がインストールされているディレクトリーです。PATH=\$PATH:/usr/lpp/java/J5.0/bin

すべての情報を 1 つのパラメーターとして渡すことができますが、コマンド行の長さに制限があるため、以下のように何らかの対話性が必要になります。

```
$ cd /etc/rdz/ssl
$ keytool -genkey -alias rdzrse -validity 3650 -keystore rdzssl.jks -storepass
rsessl -keypass rsessl
What is your first and last name?
[Unknown]: rdz rse ssl
What is the name of your organizational unit?
[Unknown]: rdz
What is the name of your organization?
[Unknown]: IBM
What is the name of your City or Locality?
[Unknown]: Raleigh
What is the name of your State or Province?
[Unknown]: NC
What is the two-letter country code for this unit?
[Unknown]: US
Is CN=rdz rse ssl, OU=rdz, O=IBM, L=Raleigh, ST=NC, C=US correct? (type "yes"
or "no")
[no]: yes
$ ls -l rdzssl.*
-rw-r--r-- 1 IBMUSER SYS1      1224 May 24 14:17 rdzssl.jks
```

上記の方法で作成した自己署名証明書は、約 10 年間有効です（うるう日はカウントしません）。これは、別名 rdzrse を使用して /etc/rdz/ssl/rdzssl.jks に保管されます。そのパスワード (rsessl) は鍵ストア・パスワードと同一であり、これは RSE の必要条件です。

結果は、以下のように -list オプションを使用して検査できます。

```
$ keytool -list -alias rdzrse -keystore rdzssl.jks -storepass rsessl -v
Alias name: rdzrse
Creation date: May 24, 2007
Entry type: keyEntry
Certificate chain length: 1
Certificate 1:
Owner: CN=rdz rse ssl, OU=rdz, O=IBM, L=Raleigh, ST=NC, C=US
Issuer: CN=rdz rse ssl, OU=rdz, O=IBM, L=Raleigh, ST=NC, C=US
Serial number: 46562b2b
Valid from: 5/24/07 2:17 PM until: 5/21/17 2:17 PM
Certificate fingerprints:
    MD5: 9D:6D:F1:97:1E:AD:5D:B1:F7:14:16:4D:9B:1D:28:80
    SHA1: B5:E2:31:F5:B0:E8:9D:01:AD:2D:E6:82:4A:E0:B1:5E:12:CB:10:1C
```

次の ssl.properties のサンプルは、server_* ディレクティブが、前に示した SAF 鍵リングのサンプルと異なることを示しています。

```
$ oedit /etc/rdz/ssl/ssl.properties
-> change: enable_ssl=true
-> uncomment and change: daemon_keydb_file=rdzssl.racf
-> uncomment and change: daemon_key_label=rdzrse
-> uncomment and change: server_keystore_file=rdzssl.jks
-> uncomment and change: server_keystore_password=rsessl
-> uncomment and change: server_keystore_label=rdzrse
-> optionally uncomment and change: server_keystore_type=JKS
```

上記の変更は SSL を使用可能に設定し、RSE サーバーに、パスワード rsessl を持つ鍵ストア rdzssl.jks 内のラベル rdzrse の下に証明書が保管されることを通知します。RSE デーモンは、引き続き SAF 準拠の鍵リングを使用しています。

付録 B. TCP/IP のセットアップ

この付録は、TCP/IP のセットアップ時、または既存のセットアップの検査時や変更時に起きる可能性があるいくつかの一般的な問題について、ユーザーを支援するためのものです。

TCP/IP 構成の詳細については、「*Communications Server IP 構成ガイド*」(SC88-8926) および「*Communications Server IP 構成解説書*」(SC88-8927) を参照してください。

ホスト名依存関係

TSO コマンド・サービスに APPC を使用する場合、Developer for System z は TCP/IP が初期化時に正しいホスト名を持っているかどうか依存します。つまり、各種の TCP/IP 構成ファイルやリゾルバー構成ファイルが正しくセットアップされていなければなりません。

TCP/IP 構成は、fekfivpt インストール検査プログラム (IVP) でテストできます。このコマンドは次のサンプルのような出力を返します (\$ は z/OS UNIX プロンプトです)。

```
$ fekfivpt
```

```
Wed Jul  2 13:11:54 EDT 2008
uid=1(USERID) gid=0(GROUP)
using /etc/rdz/rsed.envvars
```

```
-----
TCP/IP resolver configuration (z/OS UNIX search order):
-----
```

```
Resolver Trace Initialization Complete -> 2008/07/02 13:11:54.745964
```

```
res_init Resolver values:
```

```
Global Tcp/Ip Dataset = None
Default Tcp/Ip Dataset = None
Local Tcp/Ip Dataset  = /etc/resolv.conf
Translation Table      = Default
UserId/JobName          = USERID
Caller API              = LE C Sockets
Caller Mode             = EBCDIC
(L) DataSetPrefix      = TCPIP
(L) HostName            = CDFMVS08
(L) TcpIpJobName        = TCPIP
(L) DomainOrigin        = RALEIGH.IBM.COM
(L) NameServer          = 9.42.206.2
                        = 9.42.206.3
(L) NsPortAddr          = 53
(L) ResolveVia          = UDP
(*) Options NDots       = 1
(*) SockNoTestStor      = NO
(*) AlwaysWto           = NO
(*) LookUp              = DNS LOCAL
(L) ResolverTimeout     = 10
(L) ResolverUdpRetries  = 1
(L) MessageCase         = MIXED
```

```
res_init Succeeded
```

```
res_init Started: 2008/07/02 13:11:54.755363
```

```
res_init Ended: 2008/07/02 13:11:54.755371
```

```
*****
```

```
MVS TCP/IP NETSTAT CS V1R9          TCPIP Name: TCPIP          13:11:54
```

Tcpip started at 01:28:36 on 06/23/2008 with IPv6 enabled

host IP address:

hostName=CDFMVS08
hostAddr=9.42.112.75
bindAddr=9.42.112.75
localAddr=9.42.112.75

Success, addresses match

リゾルバーについて

リゾルバーは、プログラムに代わってネーム・サーバーにアクセスするクライアントとして機能し、名前からアドレス、またはアドレスから名前への解決を行います。要求側プログラムの照会を解決するために、リゾルバーは使用可能なネーム・サーバーにアクセスするか、ローカル定義 (例えば、`/etc/resolv.conf`、`/etc/hosts`、`/etc/ipnodes`、`HOSTS.SITEINFO`、`HOSTS.ADDRINFO`、または `ETC.IPNODES`) を使用するか、あるいはその両方の組み合わせを使用します。

リゾルバー・アドレス・スペースが開始されると、リゾルバー JCL プロシージャ内の `SETUP DD` カードによって指し示されている、オプションのリゾルバー・セットアップ・データ・セットが読み取られます。セットアップ情報が提供されていない場合、リゾルバーは該当するネイティブの MVS または z/OS UNIX 検索順序を使用します。その際、`GLOBALTCPIPDATA`、`DEFAULTTCPIPDATA`、`GLOBALIPNODES`、`DEFAULTIPNODES`、または `COMMONSEARCH` 情報は使用されません。

構成情報の検索順序について

TCP/IP 機能によって使用される構成ファイルの検索順序と、どのようなときにデフォルトの検索順序を環境変数、JCL、またはその他の変数によってオーバーライドできるかを理解しておくことが重要です。その知識があれば、ローカル・データ・セットと HFS ファイルの命名の標準に対応することができ、問題の診断時にも、使用されている構成データ・セットまたは HFS ファイルを知るのに役立ちます。

注意すべきもう 1 つの重要な点は、いずれかの構成ファイルに検索順序が適用された場合、最初に検出されたファイルで検索が終了するということです。したがって、絶対に検出されないファイル内に構成情報を入れた場合は、検索順序内に他のファイルが先に存在するか、またはアプリケーションが選択した検索順序内にそのファイルが含まれていないために、予期しない結果になることがあります。

構成ファイルを検索するときは、JCL プロシージャの中で `DD` ステートメントを使用するか環境変数を設定することにより、大部分の構成ファイルが置かれている場所を TCP/IP に明示的に指示できます。それ以外の場合は、「*Communications Server IP 構成ガイド*」(SC88-8926) で説明されている検索順序に基づいて、TCP/IP に構成ファイルのロケーションを動的に判別させることができます。

TCP/IP スタックの構成コンポーネントは、TCP/IP スタックの初期化時に `TCPIP.DATA` を使用して、スタックの `HOSTNAME` を判別します。その値を取得するために、z/OS UNIX 環境の検索順序が使用されます。

注: リゾルバーによって使用される TCPIP.DATA 値、およびそれらの値が読み取られた場所を判別するには、リゾルバー・トレース機能を使用します。トレースを動的に開始する方法については、「*Communications Server: IP Diagnosis Guide*」(GC31-8782) を参照してください。トレースがアクティブになった後、TSO の **NETSTAT HOME** コマンドと z/OS UNIX シェルの **netstat -h** コマンドを発行して、値を表示します。TSO および z/OS UNIX シェルからホスト名の PING を発行すると、構成されている可能性がある DNS サーバーに対するアクティビティも表示されます。

z/OS UNIX 環境で使用される検索順序

検索対象となる特定のファイルまたはテーブルは、リゾルバーの構成設定およびシステム上の所定のファイルの存在に応じて、MVS データ・セットか HFS ファイルになります。

基本リゾルバー構成ファイル

基本リゾルバー構成ファイルには、TCPIP.DATA ステートメントが入っています。このファイルは、リゾルバー・ディレクティブだけでなく、特にこのセクションで指定されている一部の構成ファイルへのアクセスを試みるときに使用されるデータ・セット接頭部 (DATASETPREFIX ステートメントの値) を判別するために、参照されます。

基本リゾルバー構成ファイルへのアクセスに使用される検索順序は、以下のとおりです。

1. GLOBALTCPIPDATA

定義されていれば、リゾルバーの GLOBALTCPIPDATA セットアップ・ステートメントの値が使用されます (330 ページの『リゾルバーについて』も参照)。検索は、追加構成ファイルについて続行されます。検索は、次のファイルが検出されると終了します。

2. 環境変数 RESOLVER_CONFIG の値

環境変数の値が使用されます。この検索は、ファイルが存在しないか他の場所で排他的に割り振られている場合、失敗します。

3. /etc/resolv.conf

4. //SYSTCPD DD カード

DD 名 SYSTCPD へ割り振られたデータ・セットが使用されます。z/OS UNIX 環境では、子プロセスは SYSTCPD DD にアクセスできません。なぜなら、SYSTCPD 割り振りは親プロセスから fork() または exec 関数呼び出しで継承されないからです。

5. userid.TCPIP.DATA

userid は、現行セキュリティ環境 (アドレス・スペース、タスク、またはスレッド) へ関連付けられているユーザー ID です。

6. jobname.TCPIP.DATA

jobname は、バッチ・ジョブの場合は JOB JCL ステートメントで指定された名前前で、開始プロシージャの場合はプロシージャ名です。

7. **SYS1.TCPPARMS(TCPDATA)**

8. **DEFAULTTCPIPDATA**

定義されていれば、リゾルバーの DEFAULTTCPIPDATA セットアップ・ステートメントの値が使用されます (330 ページの『リゾルバーについて』も参照します)。

9. **TCPIP.TCPIP.DATA**

変換テーブル

変換テーブル (EBCDIC から ASCII、および ASCII から EBCDIC) は、使用する変換データ・セットを判別するために参照されます。この構成ファイルへのアクセスに使用される検索順序は、以下のとおりです。検索順序は、最初に検出されたファイルの位置で終了します。

1. 環境変数 **X_XLATE** の値。この環境変数の値は、TSO CONVXLAT コマンドによって生成された変換テーブルの名前です。

2. **userid.STANDARD.TCPXLBIN**

userid は、現行セキュリティ環境 (アドレス・スペースまたはタスク/スレッド) へ関連付けられているユーザー ID です。

3. **jobname.STANDARD.TCPXLBIN**

jobname は、バッチ・ジョブの場合は JOB JCL ステートメントで指定された名前前で、開始プロシージャの場合はプロシージャ名です。

4. **hlq.STANDARD.TCPXLBIN**

hlq は、基本リゾルバー構成ファイル内で指定された DATASETPREFIX ステートメントがある場合は、そのステートメントの値を表します。ない場合、hlq はデフォルトの TCPIP になります。

5. テーブルが見つからない場合、リゾルバーはハードコーディングされたデフォルト・テーブルを使用します。これは、データ・セット・メンバー SEZATCPX(STANDARD) の中にリストされているテーブルと同じものです。

ローカル・ホスト・テーブル

デフォルトでは、リゾルバーは構成済みのドメイン・ネーム・サーバーがあれば、最初にそれを解決要求に使用しようとします。解決要求を満足できない場合は、ローカル・ホスト・テーブルが使用されます。リゾルバーの動作は、TCPIP.DATA ステートメントによって制御されます。

TCPIP.DATA リゾルバー・ステートメントは、ドメイン・ネーム・サーバーを使用するかどうか、およびその使用方法を定義します。LOOKUP TCPIP.DATA ステートメントは、ドメイン・ネーム・サーバーおよびローカル・ホスト・テーブルの使用方法を制御するためにも使用されます。TCPIP.DATA ステートメントの詳細については、「*Communications Server IP 構成解説書*」(SC88-8927) を参照してください。

リゾルバーは、getnetbyname API 呼び出しに対して、無条件に IPv4 固有の検索順序を使用します。sitename 情報の IPv4 固有の検索順序は、以下のとおりです。検索は、最初に検出されたファイルの位置で終了します。

1. 環境変数 **X_SITE** の値

この環境変数の値は、TSO **MAKESITE** コマンドによって作成された HOSTS.SITEINFO 情報ファイルの名前です。

2. 環境変数 **X_ADDR** の値

この環境変数の値は、TSO **MAKESITE** コマンドによって作成された HOSTS.ADDRINFO 情報ファイルの名前です。

3. **/etc/hosts**

4. **userid.HOSTS.SITEINFO**

userid は、現行セキュリティ環境 (アドレス・スペースまたはタスク/スレッド) へ関連付けられているユーザー ID です。

5. **jobname.HOSTS.SITEINFO**

jobname は、バッチ・ジョブの場合は JOB JCL ステートメントで指定された名前で、開始プロシージャの場合はプロシージャ名です。

6. **hlq.HOSTS.SITEINFO**

hlq は、基本リゾルバー構成ファイル内で指定された DATASETPREFIX ステートメントがある場合は、そのステートメントの値を表します。ない場合、hlq はデフォルトの TCPIP になります。

このセットアップ情報の Developer for System z への適用

前に述べたように、APPC を使用する場合、Developer for System z は TCP/IP が初期化時に正しいホスト名を持っているかどうか依存します。つまり、各種の TCP/IP 構成ファイルやリゾルバー構成ファイルが正しくセットアップされていなければなりません。

以下の例では、TCP/IP およびリゾルバーのいくつかの構成タスクに焦点を当てます。これは TCP/IP またはリゾルバーの完全なセットアップについて述べたものではなく、ご使用のサイトにも適用できる可能性がある、いくつかの重要な局面だけを中心に説明したものであることに注意してください。

1. 以下の JCL では、TCP/IP が SYS1.TCPPARMS(TCPDATA) を使用してスタックのホスト名を判別することが分かります。

```
//TCPIP    PROC  PARM='CTRACE(CTIEZB00)',PROF=TCPPROF,DATA=TCPDATA
//*
//* TCP/IP NETWORK
//*
//TCPIP    EXEC  PGM=EZBTCP,REGION=0M,TIME=1440,PARM=&PARMS
//PROFILE  DD   DISP=SHR,DSN=SYS1.TCPPARMS(&PROF)
//SYSTCPD  DD   DISP=SHR,DSN=SYS1.TCPPARMS(&DATA)
//SYSPRINT DD   SYSOUT=*,DCB=(RECFM=VB,LRECL=132,BLKSIZE=136)
//ALGPRINT DD   SYSOUT=*,DCB=(RECFM=VB,LRECL=132,BLKSIZE=136)
//CFGPRINT DD   SYSOUT=*,DCB=(RECFM=VB,LRECL=132,BLKSIZE=136)
//SYSOUT   DD   SYSOUT=*,DCB=(RECFM=VB,LRECL=132,BLKSIZE=136)
//CEEDUMP  DD   SYSOUT=*,DCB=(RECFM=VB,LRECL=132,BLKSIZE=136)
//SYSERROR DD   SYSOUT=*
```

2. SYS1.TCPPARMS(TCPDATA) は、ここではシステム名をホスト名にすること、およびドメイン・ネーム・サーバー (DNS) を使用しないことを示しています。すべての名前はサイト・テーブル・ルックアップによって解決されます。

```
; HOSTNAME specifies the TCP host name of this system. If not
; specified, the default HOSTNAME will be the node name specified
; in the IEFSSNxx PARMLIB member.
;
; HOSTNAME
;
; DOMAINORIGIN specifies the domain origin that will be appended
; to host names passed to the resolver. If a host name contains
; any dots, then the DOMAINORIGIN will not be appended to the
; host name.
;
DOMAINORIGIN RALEIGH.IBM.COM
;
; NSINTERADDR specifies the IP address of the name server.
; LOOPBACK (14.0.0.0) specifies your local name server. If a name
; server will not be used, then do not code an NSINTERADDR statement.
; (Comment out the NSINTERADDR line below). This will cause all names
; to be resolved via site table lookup.
;
; NSINTERADDR 14.0.0.0
;
; TRACE RESOLVER will cause a complete trace of all queries to and
; responses from the name server or site tables to be written to
; the user's console. This command is for debugging purposes only.
;
; TRACE RESOLVER
```

3. リゾルバー JCL の中で、SETUP DD ステートメントは使用されません。330 ページの『リゾルバーについて』で説明したように、これは GLOBALTCPIPDATA およびその他の変数が使用されないことを意味します。

```
//RESOLVER PROC PARMS='CTRACE(CTIRES00)'
//*
/* IP NAME RESOLVER - START WITH SUB=MSTR
/*
//RESOLVER EXEC PGM=EZBREINI,REGION=0M,TIME=1440,PARM=&PARMS
/*SETUP DD DISP=SHR,DSN=USER.PROCLIB(RESSETUP),FREE=CLOSE
```

4. RESOLVER_CONFIG 環境変数が設定されないことを想定すると、335 ページの表 51 で、リゾルバーが /etc/resolv.conf を基本構成ファイルとして使用しようとすることが分かります。

```
TCPIPJOBNAME TCPIP
DomainOrigin RALEIGH.IBM.COM
HostName CDFMVS08
```

331 ページの『z/OS UNIX 環境で使われる検索順序』で述べたように、基本構成ファイルには TCPIP.DATA ステートメントが入っています。システム名が CDFMVS08 の場合 (TCPDATA はシステム名がホスト名として使用されることを述べていました)、/etc/resolv.conf が SYS1.TCPPARMS(TCPDATA) と同期していることが分かります。DNS 定義は存在しないため、サイト・テーブル・ルックアップが使用されます。

5. 335 ページの表 51 も、デフォルトの ASCII-EBCDIC 変換テーブルを使用するために何もする必要がないことを示しています。
6. TSO MAKESITE コマンドが使用されない (X_SITE 変数および X_ADDR 変数を作成できる) と想定した場合、/etc/hosts が、名前のルックアップに使用されるサイト・テーブルになります。

```
# Resolver /etc/hosts file cdfmvs08
9.42.112.75 cdfmvs08 # CDFMVS08 Host
9.42.112.75 cdfmvs08.raleigh.ibm.com # CDFMVS08 Host
127.0.0.1 localhost
```

このファイルの最低限の内容は、現行システムに関する情報です。上記の例では、z/OS システムの IP アドレスの有効な名前として、cdfmvs08 と cdfmvs08.raleigh.ibm.com の両方を定義しています。

ドメイン・ネーム・サーバー (DNS) を使用した場合は、DNS が /etc/hosts 情報を保持し、/etc/resolv.conf および SYS1.TCPPARMS(TCPDATA) には、その DNS をシステムに対して識別するステートメントが含まれることになります。

混乱を避けるために、TCP/IP 構成ファイルとリゾルバー構成ファイルを互いに同期させておく必要があります。

表 51. リゾルバーで使用可能なローカル定義

ファイル・タイプの説明	影響する API	候補ファイル
基本リゾルバー構成ファイル	すべての API	1. GLOBALTCPIPDATA 2. RESOLVER_CONFIG 環境変数 3. /etc/resolv.conf 4. SYSTCPD DD 名 5. userid.TCPIP.DATA 6. jobname.TCPIP.DATA 7. SYS1.TCPPARMS(TCPDATA) 8. DEFAULTTCPIPDATA 9. TCPIP.TCPIP.DATA
変換テーブル	すべての API	1. X_XLATE 環境変数 2. userid.STANDARD.TCPXLBIN 3. jobname.STANDARD.TCPXLBIN 4. hlq.STANDARD.TCPXLBIN 5. リゾルバー提供の変換テーブル、SEZATCPX のメンバー STANDARD

表 51. リゾルバーで使用可能なローカル定義 (続き)

ファイル・タイプの説明	影響する API	候補ファイル
ローカル・ホスト・テーブル	endhostent endnetent getaddrinfo gethostbyaddr gethostbyname gethostent GetHostNumber GetHostResol GetHostString getnameinfo getnetbyaddr getnetbyname getnetent IsLocalHost Resolve sethostent setnetent	IPv4 1. X_SITE 環境変数 2. X_ADDR 環境変数 3. /etc/hosts 4. userid.HOSTS.xxxxINFO 5. jobname.HOSTS.xxxxINFO 6. hlq.HOSTS.xxxxINFO IPv6 1. GLOBALIPNODES 2. RESOLVER_IPNODES 環境変数 3. userid.ETC.IPNODES 4. jobname.ETC.IPNODES 5. hlq.ETC.IPNODES 6. DEFAULTIPNODES 7. /etc/ipnodes

注: 335 ページの表 51 は、「*Communications Server IP 構成ガイド*」(SC88-8926)の表の一部をコピーしたものです。完全な表については、その資料を参照してください。

ホスト・アドレスが正しく解決されない場合

TCP/IP リゾルバーでホスト・アドレスを正しく解決できない問題が生じた場合、原因として最も可能性が高いのは、リゾルバー構成ファイルが欠落しているか、不完全であることです。この問題を明確に示すものは、`lock.log` 内の以下のメッセージです。

```
clientip(0.0.0.0) <> callerip(<host IP address>)
```

これを検査するには、117 ページの『第 7 章 インストール検査』の説明に従って、`fekfivpt TCP/IP IVP` を実行します。出力のリゾルバー構成セクションは、以下のサンプルのようになります。

```
Resolver Trace Initialization Complete -> 2008/07/02 13:11:54.745964
```

```
res_init Resolver values:
Global Tcp/Ip Dataset = None
Default Tcp/Ip Dataset = None
Local Tcp/Ip Dataset = /etc/resolv.conf
Translation Table = Default
UserId/JobName = USERID
Caller API = LE C Sockets
Caller Mode = EBCDIC
```

「Local Tcp/Ip Dataset」が示すファイル (データ・セット) 内の定義が正しいことを確認してください。

このフィールドは、IP リゾルバー・ファイルに (z/OS UNIX 検索順序を使用して) デフォルト名を使用していない場合にはブランクになります。その場合は、次のステートメントを `rsed.envvars` に追加します。ここで、`<resolver file>` または `<resolver data>` は IP リゾルバー・ファイルの名前を表しています。

```
RESOLVER_CONFIG=<resolver file>
```

or

```
RESOLVER_CONFIG='<resolver data set>'
```

付録 C. INETD のセットアップ

この付録は、INETD のセットアップ時、または既存のセットアップの検査時や変更時に起きる可能性があるいくつかの一般的な問題について、ユーザーを支援するためのものです。INETD は、Developer for System z によって REXEC/SSH 機能のために使用されます。

INETD デーモンは、IP ネットワークのためのサービス管理を提供します。このデーモンはシステム負荷を軽減するために、必要な場合にだけ他のデーモンを呼び出し、そしていくつかの単純なインターネット・サービス（エコーなど）を内部的に提供します。INETD は、inetd.conf 構成ファイルを読み取って、どの追加サービスを提供するかを決定します。サービスをポートへリンクするために、ETC.SERVICES が使用されます。

inetd.conf

INETD に依存するサービスは、INETD が起動時に読み取る inetd.conf の中で定義されています。inetd.conf のデフォルトのロケーションおよび名前は、/etc/inetd.conf です。サンプル inetd.conf ファイルは、samples/inetd.conf にあります。

inetd.conf の項目には、以下の構文規則が適用されます。

- コメントはポンド記号 (#) かセミコロン (;) で始まり、行末まで続きます。
- 項目には大/小文字の区別があります。
- 項目はフィールドに依存しますが、列には依存しません。
- フィールドはスペース文字かタブ文字で分離されます。
- 項目は、以下の追加構文規則に従っていれば、複数の行にまたがってもかまいません。
 - 分割は 2 つの (スペース文字またはタブ文字によって) 分離されたワードの間で行われていなければなりません。
 - 継続行は、スペース文字またはタブ文字で始める必要があります。
 - 継続の途中にコメントを埋め込むことはできません。

各項目は 7 つの定位置フィールドからなり、次の書式に対応します。

```
service_name socket_type protocol wait_flag userid server_program
server_program_arguments
```

[ip_address:]service_name

ip_address は、ローカル IP の直後にコロン (:) を続けたものです。指定された場合、このアドレスが INADDR_ANY または現行のデフォルトの代わりに使用されます。明確に INADDR_ANY を要求するには、「*」を使用します。ip_address (またはコロン) を指定し、行にそれ以外の項目がない場合は、新しいデフォルトが指定されるまで、それが後続の行のデフォルト

トになります。service_name は、既知またはユーザー定義のサービス名です。指定する名前は、ETC.SERVICES で定義されたサーバー名の 1 つに一致する必要があります。

socket_type

stream または dgram。サービスにストリームとデータグラムどちらのソケットを使用するかを示します。

protocol[,sndbuf=n[,rcvbuf=n]]

protocol は、tcp[4|6] か udp[4|6] にすることができ、サービス名を詳しく修飾するために使用されます。サービス名とプロトコルは、両方とも ETC.SERVICES 内の項目に一致する必要があります。ただし、ETC.SERVICES 項目に「4」または「6」を含めることはできません。

sndbuf および rcvbuf は、送信バッファと受信バッファのサイズを指定します。n で表されるサイズはバイト単位とすることができ、キロバイトまたはメガバイトを示すために、それぞれ「k」または「m」を追加できます。sndbuf と rcvbuf は、どちらの順序で使用してもかまいません。

wait_flag[.max]

wait または nowait.wait はデーモンがシングル・スレッドであり、最初の要求が完了するまで別の要求がサービスを受けられないことを示します。nowait を指定した場合、INETD はストリーム・ソケット上で接続要求を受信すると、受け入れを発行します。wait を指定した場合、これがストリーム・ソケットであるときに受け入れを発行するのはサーバーの責任です。

max は、60 秒間隔でサービスを要求できるユーザーの最大数です。デフォルトは 40 です。超過した場合、サービスのポートはシャットダウンされます。

userid[.group]

userid は、fork されたデーモンを実行するユーザー ID です。このユーザー ID は、INETD ユーザー ID と異なっていてもかまいません。このユーザー ID に割り当てられる権限は、サービスの必要性によって異なります。INETD ユーザー ID は、fork されたプロセスをこのユーザー ID に切り替えるために BPX.DAEMON 権限を必要とします。

オプションの group 値は、ドット (.) によって userid から分離され、サーバーをこのユーザー ID のデフォルトと異なるグループ ID で実行できるようにします。

server_program

server_program は、サービスの絶対パス名です。例えば、/usr/sbin/rlogind は rlogind コマンドの絶対パス名です。

server_program_arguments

最大 20 個の引数。最初の引数はサーバー名です。

ETC.SERVICES

INETD は、ETC.SERVICES を使用して、サポートする必要があるサービスにポート番号とプロトコルをマップします。これは、MVS データ・セットでも z/OS UNIX ファイルでもかまいません。サンプルが SEZAINST(SERVICES) に出荷時に添付されており、これは /usr/lpp/tcpip/samples/services としても入手できます。ETC.SERVICES の検索順序は、INETD の始動方式 (z/OS UNIX か、ネイティブ MVS か) によって異なります。

サービス情報の指定には、以下の構文規則が適用されます。

- ETC.SERVICES MVS データ・セットは、固定ブロックであるか、LRECL が 56 と 256 の間の固定ブロックでなければなりません。
- ETC.SERVICES HFS ファイルでは、1 行の最大長を 256 とすることができず。
- 行の各項目は、スペース文字またはタブ文字で分離します。
- 1 つのサービスを 1 行にリストします。
- サービス名は行の最初の桁から始める必要があります。
- サービス名および別名の最大長は 32 文字です。
- 最大 35 個の別名が認識されます。
- サービス名および別名には、大/小文字の区別があります。
- コメントはポンド記号 (#) かセミコロン (;) で始まり、行末まで続きます。

各項目は 4 つの定位置フィールドからなり、次の書式に対応します。

service_name port_number/protocol aliases

service_name

既知またはユーザー定義のサービス名を指定します。

port_number

サービスに使用するソケット・ポート番号を指定します。

protocol

サービスに使用するトランスポート・プロトコルを指定します。有効な値は、tcp と udp です。

aliases 非公式のサービス名のリストを指定します。

z/OS UNIX 環境で使用される検索順序

z/OS UNIX で ETC.SERVICES にアクセスするために使用される検索順序は、以下のとおりです。検索は、最初に検出されたファイルの位置で終了します。

1. /etc/services
2. userid.ETC.SERVICES

userid は、INETD を始動するために使用されるユーザー ID です。

3. hlq.ETC.SERVICES

hlq は、基本リゾルバー構成ファイル内で指定された DATASETPREFIX ステートメントがある場合は、そのステートメントの値を表します。ない場合、hlq はデフォルトの TCPIP になります。

ネイティブ MVS 環境で使用される検索順序

ネイティブ MVS で ETC.SERVICES にアクセスするために使用される検索順序は、以下のとおりです。検索は、最初に検出されたデータ・セットの位置で終了します。

1. //SERVICES DD カード

DD ステートメント SERVICES へ割り振られたデータ・セットが使用されます。

2. userid.ETC.SERVICES

userid は、INETD を始動するために使用されるユーザー ID です。

3. jobname.ETC.SERVICES

jobname は、バッチ・ジョブの場合は JOB JCL ステートメントで指定された名前で、開始プロシージャの場合はプロシージャ名です。

4. hlq.ETC.SERVICES

hlq は、基本リゾルバー構成ファイル内で指定された DATASETPREFIX ステートメントがある場合は、そのステートメントの値を表します。ない場合、hlq はデフォルトの TCPIP になります。

注: BPXPATCH を通じて INETD を開始しても、ネイティブ MVS の検索順序が使用される結果にはなりません。BPXBATCH は z/OS UNIX 環境で開始コマンドを実行するためです。ネイティブ MVS の検索順序は、SEZALOAD(FTP) などの MVS ロード・モジュールを始動したときにのみ使用されます。

PROFILE.TCPIP ポート定義

PROFILE.TCPIP 内の PORT (または PORTRANGE) 定義と ETC.SERVICES で定義されるポートを混同しないでください。これらの定義は、用途が異なります。PROFILE.TCPIP 内に定義されているポートは、ポートが特定のサービス用に予約されているかどうかを TCPIP が調べるために使用されます。ETC.SERVICES は、INETD がポートをサービスにマップするために使用されます。

INETD は、モニター対象ポート上で要求を受信すると、inetdx と呼ばれる子プロセスを (要求されたサービスを使用して) fork します。ここで、inetd は INETD のジョブ名 (始動方式によって異なります) で、x は 1 桁の番号です。

これはポート予約を複雑にするので、INETD モニター対象ポートが PROFILE.TCPIP 内で予約されている場合は、z/OS UNIX カーネル・アドレス・スペース用の開始 JCL プロシージャの名前を使用して、ほぼすべてのプロセスをそのポートにバインドできるようにする必要があります。この名前には OMVS がよく使用されます。ただし、BPXPRMxx parmlib メンバーの STARTUP_PROC パラメーターで別の名前を明示的に指定した場合は除きます。

以下のリストは、アプリケーション実行環境が与えられた場合に、ジョブ名を判別する方法を説明したものです。

- バッチから実行されたアプリケーションはバッチ・ジョブ名を使用します。

- MVS オペレーター・コンソールから開始されたアプリケーションは、開始プロシージャー (STC) 名をジョブ名として使用します。
- TSO ユーザー ID から実行されたアプリケーションは、TSO ユーザー ID をジョブ名として使用します。
- z/OS シェルから実行されたアプリケーションは通常、ログオン・ユーザー ID に 1 文字の接尾部を付加したジョブ名を持っています。
- 権限があるユーザーは z/OS シェルからアプリケーションを実行し、`_BPX_JOBNAME` 環境変数を使用してジョブ名を設定できます。この場合、環境変数に指定された値が、ジョブ名です。
- UNIX システム・サービス・カーネル・アドレス・スペース用の開始 JCL プロシージャーの名前を使用して、`bind()` ソケット API のほとんどすべての呼び出し元 (Pascal API のユーザーを除く) がポートにバインドできるようにすることができます。この名前には OMVS がよく使用されます。ただし、BPXPRMxx parmlib メンバーの `STARTUP_PROC` パラメーターで別の名前を明示的に指定した場合は除きます。
- INETD によって開始された z/OS UNIX アプリケーションは、INETD サーバーのジョブ名を使用します。

注: お勧めはできませんが、ETC.SERVICES 内で定義するポートは、`PROFILE.TCPIP` 内のサービスの予約済みポート番号と異なってもかまいません。

/etc/inetd.pid

INETD プロセスは、一時ファイル `/etc/inetd.pid` を作成し、これには現在実行中の INETD デーモンの PID (プロセス ID) が入っています。この PID 値は、INETD プロセスから発信された syslog レコードを識別するために使用されるほか、kill など、PID を必要とするコマンドに PID 値を提供するためにも使用されます。また、複数の INETD プロセスがアクティブにならないようにするロック・メカニズムとしても使用されます。

始動

INETD の z/OS UNIX 実装は、デフォルトでは `/usr/sbin/inetd` に配置され、2 つの非定位置の、オプションの始動パラメーターをサポートしています。

`/usr/sbin/inetd [-d] [inetd.conf]`

-d デバッグ・オプション。デバッグ出力は `stderr` に書き込まれ、これは、`syslogd` デーモンによってファイルへ送ることができます。`syslogd` の詳細については、「*Communications Server IP 構成ガイド*」(SC88-8926) を参照してください。この方法で始動された INETD は、サービスを開始するために子プロセスを fork しないことに注意してください。

inetd.conf

構成ファイル。デフォルト値は `/etc/inetd.conf` です。

INETD は IPL 時に始動する必要があります。これを行う最も一般的な方法は、INETD を `/etc/rc` または `/etc/inittab` (z/OS 1.8 以上のみ) から始動することで

す。また、BPXBATCH を使用してジョブまたは開始タスクから始動するか、適切な権限を持つユーザーのシェル・セッションから始動することもできます。

/etc/rc

INETD は、z/OS UNIX 初期化シェル・スクリプト /etc/rc から始動された場合、z/OS UNIX 検索順序を使用して ETC.SERVICES を検出します。サンプル /etc/rc ファイルが /samples/rc として出荷時に添付されています。以下のサンプル・コマンドを使用して、INETD を始動できます。

```
# Start INETD
_BPX_JOBNAME='INETD' /usr/sbin/inetd /etc/inetd.conf &
sleep 5
```

/etc/inittab

z/OS 1.8 以上には、z/OS UNIX の初期化時にコマンドを発行するための代替の方式、/etc/inittab が用意されています。/etc/inittab を使用すると、終了時に自動的にプロセスを再始動する respawn パラメーターを定義できます (15 分以内に、2 番目の再始動について WTOR がオペレーターへ送信されます)。INETD は、/etc/inittab から始動された場合、z/OS UNIX 検索順序を使用して ETC.SERVICES を検出します。サンプル /etc/inittab が /samples/inittab として出荷時に添付されています。次のサンプル・コマンドを使用して、INETD を始動できます。

```
# Start INETD
inetd::respfrk:/usr/sbin/inetd /etc/inetd.conf
```

注: サンプルに使用されている respfrk パラメーターは、RSE も含め、すべての fork されたプロセスに respawn 属性を転送することに留意してください。クライアントが接続を閉じると、respawn は再び接続を開始します。RSE サーバーは、後で再び、タイムアウトのために終了します。inittab の詳細については、「*UNIX System Services 計画*」(GA88-8639) を参照してください。

BPXBATCH

BPXBATCH 始動方式は、開始タスクとユーザー・ジョブのどちらにも使用できます。INETD はバックグラウンド・プロセスなので、INETD を始動する BPXBATCH ステップは、始動後数秒以内に終了します。INETD は、BPXBATCH によって始動された場合、z/OS UNIX 検索順序を使用して ETC.SERVICES を検出します。以下のコード・サンプルにリストされている JCL は、INETD を始動するサンプル・プロシージャです (KILL ステップは、アクティブな INETD プロセスがあれば、それを除去します)。

```
//INETD    PROC PRM=
//*
//KILL      EXEC PGM=BPXBATCH,REGION=0M,
//          PARM='SH ps -e | grep inetd | cut -c 1-10 | xargs -n 1 kill'
//*
//INETD      EXEC PGM=BPXBATCH,REGION=0M,
//          PARM='PGM /usr/sbin/inetd &PRM'
//STDERR    DD SYSOUT=*
//* STDIN, STDOUT and STDENV are defaulted to /dev/null
//*
```

図 64. INETD 始動 JCL

注:

- STDIN、STDOUT、および STDERR は、割り振る場合、z/OS UNIX ファイルでなければなりません。STDENV は、MVS データ・セットでも z/OS UNIX ファイルでもかまいません。z/OS 1.7 以降、SYSOUT を STDOUT および STDERR に割り当てることができます。BPXBATCH の詳細については、「*UNIX System Services コマンド解説書*」(SA88-8641) を参照してください。
- inetd.conf は、INETD を BPXBATCH によって始動する場合、MVS データ・セットまたはメンバーとすることができます。その場合は、次のサンプルのような PARM ステートメントをコーディングします (単一引用符 (') のみを使用します)。

```
// PARM='PGM /usr/sbin/inetd //'SYS1.TCPPARMS(INETCONF)'' &PRM'
```

シェル・セッション

INETD は、シェル・セッションから始動された場合、ETC.SERVICES を検出するために z/OS UNIX 検索を使用します。以下のサンプル・コマンドを使用して、(十分な権限を持つユーザーは) INETD の停止と始動を行うことができます (# は z/OS UNIX プロンプトです)。

```
# ps -e | grep inetd
7 ?          0:00 /usr/sbin/inetd
# kill 7
# _BPX_JOBNAME='INETD' /usr/sbin/inetd &
```

注: この方式は、初期始動にはお勧めできません。/etc/rc または /etc/inittabの方が、z/OS UNIX の初期化時に実行されるので、適切です。

セキュリティ

INETD は z/OS UNIX プロセスなので、セキュリティ・ソフトウェア内に INETD へ関連付けられたユーザー ID 用の有効な OMVS 定義が必要です。ユーザー ID に対して UID、HOME、および PROGRAM を設定し、ユーザーのデフォルト・グループに対して GID を設定する必要があります。INETD が /etc/rc または /etc/inittab によって始動された場合、ユーザー ID は z/OS UNIX カーネルから継承され、デフォルトは OMVSKERN です。

```
ADDGROUP OMVSGRP OMVS(GID(1))
ADDUSER OMVSKERN DFLTGRP(OMVSGRP) NOPASSWORD +
        OMVS(UID(0) HOME('/') PROGRAM('/bin/sh'))
```


INETD は、setuid() などの関数へのアクセスを必要とするデーモンです。したがって、INETD の始動に使用するユーザー ID は、FACILITY クラス内の BPX.DAEMON プロファイルへの READ アクセス権を必要とします。このプロファイルが定義されていない場合は、UID 0 が必要です。

```
PERMIT BPX.DAEMON CLASS(FACILITY) ACCESS(READ) ID(OMVSKERN)
```

INETD ユーザー ID には、inetd プログラム (/usr/sbin/inetd) の EXECUTE 権限、inetd.conf および ETC.SERVICES ファイルへの READ アクセス権、/etc/inetd.pid への WRITE アクセス権も必要です。UID 0 を使用せずに INETD を実行したい場合は、UNIXPRIV クラス内の SUPERUSER.FILESYS プロファイルに対する CONTROL アクセス権を与えて、z/OS UNIX ファイルに必要な許可を提供することができます。

デーモン権限を必要とするプログラムは、FACILITY クラス内で BPX.DAEMON が定義されている場合、プログラムで制御されている必要があります。これは、デフォルトの INETD プログラム (/usr/sbin/inetd) の場合は、既にそうになっていますが、コピーまたはカスタム・バージョンを使用する場合は手動で設定する必要があります。z/OS UNIX ファイルをプログラムで制御されるようにするには、**extattr +p** コマンドを使用します。MVS ロード・モジュールをプログラムで制御されるようにするには、RACF PROGRAM クラスを使用します。

INETD をシェル・セッションの中から再始動する必要があるシステム・プログラマーは、自分の許可を使用して INETD を始動します。したがって、正規の INETD ユーザー ID と同じ許可リストを持っている必要があります。それに加えて、INETD プロセスをリストおよび停止する許可も必要です。これは、複数の方法で実現できます。

- UID 0

これは、「人間」のユーザー ID には推奨できません。その理由は、z/OS UNIX に関連した制約事項が存在しないからです。

- FACILITY クラス内の BPX.SUPERUSER プロファイルへの READ アクセス権

ユーザーが **su** コマンドによって UID 0 になることを許可します。これは、推奨できるセットアップです。

- 必要な権限を対象とした個別プロファイルへのアクセス権

- UNIXPRIV クラス内の SUPERUSER.PROCESS.GETPSENT への READ アクセス権 (**ps** コマンド用)
- UNIXPRIV クラス内の SUPERUSER.PROCESS.KILL への READ アクセス権 (**kill** コマンド用)
- FACILITY クラス内の BPX.JOBNAME への READ アクセス権 (**_BPX_JOBNAME** 環境変数用)

extattr コマンドおよび **su** コマンドの詳細については、「*UNIX System Services コマンド解説書*」(SA88-8641) を参照してください。UNIXPRIV クラスの詳細、および FACILITY クラス内の BPX.* プロファイルの詳細については、「*UNIX System Services 計画*」(GA88-8639) を参照してください。OMVS セグメント定義および PROGRAM クラスの詳細については、「*Security Server RACF セキュリティー管理者のガイド*」(SA88-8613) を参照してください。

Developer for System z の要件

Developer for System z は、REXEC や SSH の管理に関して、INETD に依存しています。また、上記の INETD セットアップに加えて、追加の要件を課す場合があります。

REXEC (または SSH) は、109 ページの『(オプション) REXEC (または SSH) の使用』で説明されているように、次の 2 つの目的で使用されます。

- z/OS UNIX サブプロジェクトでのリモート (ホスト・ベースの) アクション
- 代替 RSE サーバー始動方式

z/OS UNIX サブプロジェクトでのリモート・アクションに、特別な設定は必要ありません。しかし、代替の RSE 始動方式には、特別な設定が必要です。

INETD

INETD で RSE サーバーを始動するためには、INETD の環境設定 (これはプロセスの開始時に受け渡されます) と、INETD のユーザー ID の権限が正しく設定されている必要があります。

- INETD が BPXBATCH を使用して JCL によって始動される場合、領域サイズは 0 でなければなりません。
- INETD が TSO/OMVS シェル・セッションから始動される場合、TSO 領域サイズは 2096128 以上でなければなりません。
- INETD が /etc/rc または /etc/inittab によって始動される場合は、SYS1.PROCLIB(BPX0INIT) の領域サイズが使用され、そのデフォルトは 0 です。

REXEC (または SSH)

クライアントがポート 512 (または 22) に接続したときに INETD によって始動される REXEC (または SSH) デーモンを使用して、認証が実行され、RSE サーバーが始動され、それ以降の通信用のポート番号がクライアントへ返されます。そのためには、REXEC (SSH) デーモンへ (inetd.conf 内で) 割り当てられたユーザー ID に以下の権限が必要です。

- セキュリティー・ソフトウェア内の有効な OMVS 定義。UID、HOME、および PROGRAM が、そのユーザーのデフォルト・グループの GID とともに、設定されていなければなりません。
- FACILITY クラス内の BPX.DAEMON プロファイルへの READ アクセス権
- Developer for System z インストール・ディレクトリー (デフォルトは /usr/lpp/rdz/*) への READ および EXECUTE アクセス権。
- Developer for System z 構成ディレクトリー (デフォルトは /etc/rdz/*) への READ および EXECUTE アクセス権。

付録 D. APPC のセットアップ

この付録は、APPC (拡張プログラム間通信機能) のセットアップ時、または既存のセットアップの検査時や変更時に起きる可能性があるいくつかの一般的な問題について、ユーザーを支援するためのものです。

以下に述べる APPC 管理および parmlib メンバーの詳細については、「*MVS 計画: APPC/MVS 管理*」(SA88-8571) および「*MVS 初期設定およびチューニング解説書*」(SA88-8564) を参照してください。

これは APPC の完全なセットアップについて述べたものではなく、ご使用のサイトにも適用できる可能性がある、いくつかの重要な局面のみを中心にした説明であることに注意してください。

メンバー SYS1.SAMPLIB(ATBALL) には、SYS1.SAMPLIB 内のすべての APPC 関連 (サンプル) メンバーのリストと説明が入っています。

VSAM

APPC/MVS は、構成データを以下の SYS1.PARMLIB メンバーおよび 2 つの VSAM データ・セットに保管します。

- トランザクション・プログラム (TP) VSAM データ・セット (デフォルト名 SYS1.APPCTP) には、z/OS プログラムのスケジューリングおよびセキュリティ情報が入っています。
- サイド情報 (SI) VSAM データ・セット (デフォルト名 SYS1.APPCSI) には、z/OS ローカル TP および APPC/MVS サーバーによって使用されるシンボリック宛先名の翻訳が入っています。

TP は APPC を使用して、同じシステム上または別のシステム上の TP と通信し、リソースにアクセスするアプリケーション・プログラムです。Developer for System z 用の APPC セットアップは、FEKFRSRV と呼ばれる新しい TP をアクティブにします。この TP は、TSO コマンド・サービスと呼ばれます。

次のジョブはサンプル・メンバー SYS1.SAMPLIB(ATBTPVSM) と SYS1.SAMPLIB(ATBSIVSM) の連結であり、APPC VSAM の定義に使用できます。

```

//APPCVSAM JOB <job parameters>
//*
/* CAUTION: This is neither a JCL procedure nor a complete job.
/* Before using this sample, you will have to make the following
/* modifications:
/* 1. Change the job parameters to meet your system requirements.
/* 2. Change ***** to the volume that will hold the APPC VSAMs.
/*
//TP      EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN   DD *
        DEFINE CLUSTER (NAME(SYS1.APPCTP) -
                        VOLUME(*****)) -
                        INDEXED REUSE -
                        SHAREOPTIONS(3 3) -
                        RECORDSIZE(3824 7024) -
                        KEYS(112 0) -
                        RECORDS(300 150)) -
        DATA      (NAME(SYS1.APPCTP.DATA)) -
        INDEX      (NAME(SYS1.APPCTP.INDEX))
/*
//SI      EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN   DD *
        DEFINE CLUSTER (NAME(SYS1.APPCSI) -
                        VOLUME(*****)) -
                        INDEXED REUSE -
                        SHAREOPTIONS(3 3) -
                        RECORDSIZE(248 248) -
                        KEYS(112 0) -
                        RECORDS(50 25)) -
        DATA      (NAME(SYS1.APPCSI.DATA)) -
        INDEX      (NAME(SYS1.APPCSI.INDEX))
/*

```

図 65. APPC VSAM を作成するための JCL

VTAM

APPC はシステム・ネットワーク体系 (SNA) LU 6.2 プロトコルの実装です。SNA は、さまざまな物理および論理 SNA コンポーネント (論理装置 (LU) など) を定義する形式とプロトコルを提供します。LU 6.2 は、特にアプリケーション・プログラム間の通信を処理するように設計された論理装置タイプです。

MVS 上で SNA を使用するためには、VTAM (仮想記憶通信アクセス方式) をインストールし、構成する必要があります。APPC システム・タスクを使用するには、VTAM をアクティブにしておく必要があります。

VTAM セットアップの APPC 固有の部分は、以下の 3 つのステップからなっています。

1. SYS1.SAMPLIB(ATBLJOB) を使用して、使用するモード名 (例えば、APPCHOST) を VTAM に対して定義し、SYS1.SAMPLIB(ATBLMODE) を SYS1.VTAMLIB 内へアセンブルおよびリンク・エディットします。詳細については、メンバー SYS1.SAMPLIB(ATBLMODE) を参照してください。

2. サンプル・メンバー SYS1.SAMPLIB(ATBAPPL) を SYS1.VTAMLST 連結内のデータ・セットにコピーすることにより、APPC/MVS を VTAM アプリケーションとして定義します。詳細については、メンバー SYS1.SAMPLIB(ATBAPPL) を参照してください。
3. コンソール・コマンド **v net,act,id=atbappl** を使用して、新規に定義したアプリケーションをアクティブにします (ここで、net は使用する VTAM 開始タスクの名前です)。コンソール・コマンド **d net,appls** を使用して、アプリケーションがアクティブであることを確認します。VTAM の始動時にメンバーをアクティブにしたい場合は、SYS1.VTAMLST(ATCCONxx) にメンバー名を追加します。

サンプル・メンバー SYS1.SAMPLIB(ATBAPPL) 内で使用されている MVSLU01 の ACBNAME は、サイトの標準に合わせて変更できますが、SYS1.PARMLIB(APPCPMxx) メンバー内の定義に一致する必要があります。

```
MVSLU01 APPL  ACBNAME=MVSLU01,      C
                APPC=YES,            C
                AUTOSSES=0,          C
                DDRAINL=NALLOW,     C
                DLOGMOD=APPCHOST,    C
                DMINWNL=5,           C
                DMINWNR=5,           C
                DRESPL=NALLOW,      C
                DSESLIM=10,          C
                LMDENT=19,            C
                MODETAB=LOGMODES,    C
                PARSESS=YES,         C
                SECACPT=CONV,        C
                SRBEXIT=YES,         C
                VPACING=1
```

図 66. SYS1.SAMPLIB(ATBAPPL)

VTAM の詳しい構成方法については、「*Communications Server IP SNA ネットワーク・インプリメンテーション・ガイド*」(SC88-8928) を参照してください。

SYS1.PARMLIB(APPCPMxx)

システム間の会話フローを使用可能にし、サポートするには、セッションをバインドできる論理装置 (LU) 同士をサイトで定義する必要があります。APPC/MVS 処理を行うには、APPC 処理が単一システム上に残留する場合でも、事前にサイトで 1 つ以上の LU を定義しておく必要があります。LU は、SYS1.PARMLIB(APPCPMxx) 内で行われる定義の一部です。

TSO コマンド・サービスを使用するには、APPC が、インバウンドとアウトバウンドの両方の要求を処理できる基本 LU を持つようにセットアップされている必要があります。

LU 定義は、SYS1.PARMLIB(APPCPMxx) メンバーに追加する必要があり、BASE パラメーターと SCHED(ASCH) パラメーターを含んでいる必要があります。APPCPMxx メンバーは、使用するトランザクション・プロファイル (TP) とサイド情報 (SI) の VSAM データ・セットも指定する必要があります。

以下のコード・サンプルは、TSO コマンド・サービスに使用できる
SYS1.PARMLIB(APPCPMxx) メンバーです。

```
LUADD
  ACBNAME(MVSLU01)
  BASE
  SCHED(ASCH)
  TPDATA(SYS1.APPCTP)
SIDEINFO DATASET(SYS1.APPCSI)
```

図 67. SYS1.PARMLIB(APPCPMxx)

システムに複数の LU 名があるときは、システムがどの LU を BASE LU として
選択するかに応じて、変更が必要になる場合があります。システムの BASE LU
は、以下のようにして決定されます。

1. システム基本 LU は、NOSCHED と BASE の両方のパラメーターを含んでいる最
後の LUADD ステートメントによって表されます。このタイプのシステム基本
LU は、アクティブなトランザクション・スケジューラーがないときに、アウト
バウンド要求を処理できます。
2. NOSCHED と BASE の両方を含んでいる LUADD ステートメントがない場合は、
BASE パラメーターを含み、ASCH を APPC/MVS トランザクション・スケジュー
ラーとして指定している最後の LUADD ステートメントによってシステム基本
LU が表されます。その場合の APPC/MVS トランザクション・スケジューラー
の指定は、SCHED(ASCH) をコーディングするか、SCHED パラメーターをまったく
コーディングしないことによって行います (ASCH は SCHED のデフォルト値で
す)。

注: オペレーター・コマンド **D APPC,LU,ALL** はアクティブなすべての LU 定
義を示し、基本 LU にマークを付けます。

ご使用のシステムに BASE パラメーターおよび NOSCHED パラメーターを持つ LU
がある場合は、その LU が BASE LU として使用されるはずですが、しかし、その
LU には FEKFRSRV トランザクションへの要求を処理するトランザクション・スケ
ジューラーがないために、TSO コマンド・サービスは機能しません。その LU を変
更して NOSCHED パラメーターを除去できない場合は、BASE と SCHED(ASCH) を持つ
LU に対して、次のような rsed.envvars 環境変数 `_FEKFSCMD_PARTNER_LU` を設定
できます。

```
_FEKFSCMD_PARTNER_LU=MVSLU01
```

rsed.envvarsの詳細については、35 ページの『rsed.envvars、RSE 構成ファイル』
を参照してください。

SYS1.PARMLIB(ASCHPMxx)

APPC/MVS トランザクション・スケジューラー (デフォルト名は ASCH) は、会話を
求めるインバウンド要求に対する応答として、トランザクション・プログラムを開
始およびスケジュールします。メンバー SYS1.PARMLIB(ASCHPMxx) は、例えば、ト
ランザクション・クラス定義などを使用して、その機能を制御します。

TSO コマンド・サービスに使用される APPC トランザクション・クラスには、Developer for System z の各ユーザーに 1 つずつイニシエーターを許可できるだけの十分な数の APPC イニシエーターがあることが必要です。

TSO コマンド・サービスを使用するには、OPTIONS セクションと TPDEFAULT セクションでデフォルトの仕様を指定する必要があります。

以下のコード・サンプルは、TSO コマンド・サービスに使用できる SYS1.PARMLIB(ASCHPMxx) メンバーです。

```
CLASSADD
  CLASSNAME(A)
  MAX(20)
  MIN(1)
  MSGLIMIT(200)

OPTIONS
  DEFAULT(A)

TPDEFAULT
  REGION(2M)
  TIME(5)
  MSGLEVEL(1,1)
  OUTCLASS(X)
```

図 68. SYS1.PARMLIB(ASCHPMxx)

注:

- IBM サポートは、デバッグの目的で、ログ・ファイルに書き込まれる出力の量を増やすために、MSGLIMIT の値を大きくするように求める場合があります。
- オペレーター・コマンド **D ASCH,ALL** はアクティブなすべての APPC トランザクション・スケジューラー・クラスを示します。

APPC の変更のアクティブ化

この時点で、上記の各ステップで説明した構成の変更をアクティブにすることができます。そのためには、以下のように、環境に応じてさまざまな方法があります。

- APPC がまだアクティブでない場合。以下のコンソール・コマンドを入力して、APPC/MVS を始動します (ここで、xx は関連する SYS1.PARMLIB メンバーの最後の 2 文字です)。

1. S APPC,SUB=MSTR,APPC=xx
2. S ASCH,SUB=MSTR,ASCH=xx

システム始動時にこれらのコマンドを開始するには、これらを SYS1.PARMLIB(COMMNDxx) に追加します。

- APPC が既にアクティブである場合。APPC は、以下のコンソール **SET** コマンドを使用することにより、SYS1.PARMLIB メンバーを動的に再ロードできます (ここで、xx は関連する SYS1.PARMLIB メンバーの最後の 2 文字です)。

1. SET APPC=xx
2. SET ASCH=xx

コンソール・コマンド **D APPC** および **D ASCH** を使用して、APPC セットアップを検査できます。上記のコンソール・コマンドの詳細については、「*MVS システム・コマンド*」(GC88-6592) を参照してください。

TSO コマンド・サービス・トランザクションの定義

APPC/MVS をアクティブにした後、111 ページの『(オプション) TSO コマンド・サービス用の APPC トランザクション』の説明に従って Developer for System z TSO コマンド・サービスを定義できます。

説明されている APPC トランザクションの定義方法は、FEK.#CUST.JCL(FEKAPPC) をカスタマイズして実行依頼することです。

APPC トランザクションは、APPC ISPF インターフェースを通じて対話的に定義することもできます。これについては、ホワイト・ペーパーに説明があります。このホワイト・ペーパーでは、ユーザー固有のアカウント情報を収集するための APPC トランザクションのセットアップ方法も説明されています。

「*APPC and WebSphere Developer for System z*」(SC23-5885-00) ホワイト・ペーパーは、Developer for System z インターネット・ライブラリー (<http://www-306.ibm.com/software/awdtools/rdz/library/>) で入手できます。

注: TSO コマンド・サービスを開始するために APPC によって使用されるトランザクション・プログラム (TP) JCL は、Developer for System z バージョン 7.1 で変更されました。ホワイト・ペーパーの説明に従って TP を定義する場合は、次の例のように、PARM 行に NESTMACS キーワードを追加する必要があります。

```
// PARM='ISPSTART CMD(%FEKFRSRV TIMEOUT=60) NEWAPPL(ISR) NESTMACS'
```

(オプション) 代替セットアップ・オプション

Developer for System z は代替の APPC および VTAM セットアップ・オプションをサポートしており、このセクションでは、その一部について説明します。

代替トランザクション名

TSO コマンド・サービスのデフォルトのトランザクション名は、111 ページの『(オプション) TSO コマンド・サービス用の APPC トランザクション』で説明されているように、FEKFRSRV です。同じセクションの説明にあるように、この名前は APPC に対してトランザクションを定義するときに変更できます。

APPC でトランザクション名を変更することは、35 ページの『rsed.envvars、RSE 構成ファイル』で説明されているように、rsed.envvars で `_FEKFSCMD_TP_NAME_` に新しい名前を割り当てる必要があることを意味します。

複数の LU

APPC は、プログラム (パートナー・ノード) とホスト上のプログラム (ローカル・ノード) との対話を可能にする通信プロトコルです。Developer for System z を使用した場合、パートナー・ノード (TSO コマンド・サーバー) とローカル・ノード

(RSE サーバー) の両方が、同じ z/OS システム上でアクティブになります。また、デフォルトでは、両方が同じ (BASE) LU 定義を使用して互いに通信します。

35 ページの『rsed.envvars、RSE 構成ファイル』で説明されているように、rsed.envvars の `_FEKFSCMD_PARTNER_LU_` ディレクティブの中で、TSO コマンド・サービスの代替パートナー LU 名を指定できます。ローカル LU 名は、常に有効な BASE LU である (BASE キーワードと SCHED キーワードを持つ) ことが必要なので、変更できないことに注意してください。

LU セキュリティー

VTAM は、セキュアな APPC セットアップをサポートしており、パートナーとローカル LU の間の通信をセキュリティー・ソフトウェアに対して定義する必要があります。

これは、`VERIFY=REQUIRED` をローカル (BASE) LU の VTAM 定義に追加することによってアクティブになります。セキュリティー定義は APPCLU クラス内で行う必要があります。これについては、「MVS 計画: APPC/MVS 管理」(SA88-8571) で説明されています。

このセットアップが VTAM 内でアクティブであり、セキュリティー・ソフトウェアでのセットアップが完了していない場合、TSO コマンド・サービスとの通信は初期化に失敗し、システム・ログに VTAM が接続のセットアップを拒否したことを示すメッセージが何も残されないので注意してください。APPC IVP テスト (fekfivpa) は「戻りコード 1 - 割り振りの失敗、再試行なし (Return code 1 - Allocate Failure no retry)」で失敗します。

付録 E. 必要条件

この付録では、このバージョンの Developer for System z でのホストの前提条件および相互前提条件をリストします。

必須の必要条件とオプションの必要条件の最新リストについては、Developer for System z オンライン・ライブラリー (<http://www-01.ibm.com/software/awdtools/rdz/library/>) で「*Rational Developer for System z Prerequisites*」(SC23-7659) を参照してください。(日本語版:「*Rational Developer for System z 前提条件*」(SC88-4704) も出版されています。)

このセクションでリストする製品は、すべて本書の発行時に使用可能なものです。関連する Developer for System z 機能を使用する際に、選択した製品が引き続き使用可能かどうかを確認するには、IBM Software Support Lifecycle Web サイト (<http://www.ibm.com/software/support/lifecycle/>) を参照してください。

z/OS ホストの前提条件

Developer for System z を使用するには、適切な前提条件を備えた次の環境が必要です。

z/OS

以下のいずれかのレベルがホストにインストールされている必要があります。

プログラム番号	プロダクト名	必要な PTF またはサービス・レベル
5694-A01	z/OS v 1.11	ISPF: • APAR OA29489 (TSO/ISPF クライアント・ゲートウェイ) PTF UA51713 TCP/IP: • PTF またはサービス・レベルは必要ありません
5694-A01	z/OS v 1.10	ISPF: • APAR OA29489 (TSO/ISPF クライアント・ゲートウェイ) PTF UA51712 TCP/IP: • APAR PK74282 (CSM 固定ストレージの拡張) PTF UK41810

プログラム 番号	プロダクト名	必要な PTF またはサービス・レベル
5694-A01	z/OS v 1.9	ISPF: <ul style="list-style-type: none"> • APAR OA29489 (TSO/ISPF クライアント・ゲートウェイ) PTF UA51687 TCP/IP: <ul style="list-style-type: none"> • APAR PK74282 (CSM 固定ストレージの拡張) PTF UK41812
5694-A01	z/OS v 1.8	ISPF: <ul style="list-style-type: none"> • APAR OA20345 (ネストされたコマンド出力) PTF UA33575 • APAR OA20449 (NESTMACS サポートを追加) PTF UA34052 • APAR OA29489 (TSO/ISPF クライアント・ゲートウェイ) PTF UA51686 TCP/IP: <ul style="list-style-type: none"> • APAR PK74282 (CSM 固定ストレージの拡張) PTF UK41811

関連する製品 Web サイトは、次のとおりです。

<http://www-03.ibm.com/systems/z/os/zos/>

注:

1. z/OS UNIX サブプロジェクトのリモート (ホスト・ベースの) アクションを実行するには、ホスト上で z/OS UNIX バージョンの REXEC または SSH がアクティブであることが必要です。
2. z/OS には以下のコンポーネントが含まれており、これらをインストールし、構成し、操作可能にしておく必要があります。
 - 対話式システム生産性向上機能 (ISPF)
 - <http://www-01.ibm.com/software/awdtools/ispf/>
 - 言語環境プログラム
 - <http://www-03.ibm.com/servers/eserver/zseries/zos/le/>
 - RACF または同等のセキュリティー製品
 - <http://www-03.ibm.com/servers/eserver/zseries/zos/racf/>
 - IBM Communications Server の VTAM コンポーネント
 - <http://www-01.ibm.com/software/network/commsserver/zos/>
 - IBM Communications Server の IP サービス・コンポーネント
 - <http://www-01.ibm.com/software/network/commsserver/zos/>
 - パインダー
 - APPC (オプション)

注:

- APPC は、ご使用のホスト・システム上で ISPF APAR OA29489 のサービスが使用可能でない場合は、必須の前提条件です。
- APPC は ISPF クライアント・ゲートウェイ機能で置き換えることができます。この機能は ISPF for z/OS 1.10 に組み込まれており、適切な PTF を適用した ISPF for z/OS 1.8 および 1.9 で使用可能です。

SMP/E

Developer for System z をインストールするためには、以下のいずれかのレベルがインストールされている必要があります。

プログラム番号	プロダクト名	必要な PTF またはサービス・レベル
5655-G44	IBM System Modification Program Extended (SMP/E) for z/OS v 3.5	PTF またはサービス・レベルは必要ありません
5655-G44	IBM System Modification Program Extended (SMP/E) for z/OS v 3.4	PTF またはサービス・レベルは必要ありません

関連する製品 Web サイトは、次のとおりです。

<http://www-03.ibm.com/systems/z/os/zos/smpe/>

SDK for z/OS Java 2 Technology Edition

リモート・システム・エクスプローラー (RSE) を使用するアプリケーションをサポートするためには、以下のいずれかのレベルがホストにインストールされている必要があります。

プログラム番号	プロダクト名	必要な PTF またはサービス・レベル
5655-R32	IBM 64 ビット SDK for z/OS、Java 2 Technology Edition、v 6.0	サービス・リリース 7
5655-R31	IBM 31-bit SDK for z/OS、Java 2 Technology Edition v 6.0	サービス・リリース 7
5655-N99	IBM 64 ビット SDK for z/OS、Java 2 Technology Edition、v 5.0	サービス・リリース 11
5655-N98	IBM 31-bit SDK for z/OS、Java 2 Technology Edition v 5.0	サービス・リリース 11

関連する製品 Web サイトは、次のとおりです。

<http://www.ibm.com/servers/eserver/zseries/software/java/>

注: 64 ビット版 Java を使用する場合は、Developer for System z APAR PM07305 用の PTFを適用する必要があります。この PTF は、Developer for System z 推奨サービスのページ (<http://www-01.ibm.com/support/docview.wss?rs=2294&context=SS2QJ2&uid=swg27006335>)で入手できます。

z/OS ホストの相互前提条件

Developer for System z の特定のフィーチャーをサポートするには、このセクションにリストされている製品と記載されているその他のソフトウェアが必要です。
Developer for System z ワークステーション・クライアントは、これらの必要条件なしに正常にインストールできます。ただし、該当するフィーチャーを設計どおりに機能させるには、記載されているホストの必要条件をインストールし、実行時に操作可能になるようにしておく必要があります。

z/OS

プログラム番号	プロダクト名	必要な PTF またはサービス・レベル
5694-A01	z/OS v 1.11	HLASM PTF またはサービス・レベルは必要ありません XL C/C++ PTF またはサービス・レベルは必要ありません SCLM PTF またはサービス・レベルは必要ありません LE (PL/I) PTF またはサービス・レベルは必要ありません TN3270 PTF またはサービス・レベルは必要ありません
5694-A01	z/OS v 1.10	HLASM PTF またはサービス・レベルは必要ありません XL C/C++ PTF またはサービス・レベルは必要ありません SCLM PTF またはサービス・レベルは必要ありません LE (PL/I) PTF またはサービス・レベルは必要ありません TN3270 PTF またはサービス・レベルは必要ありません

プログラム 番号	プロダクト 名	必要な PTF またはサービス・レベル
5694-A01	z/OS v 1.9	HLASM PTF またはサービス・レベルは必要ありません XL C/C++ PTF またはサービス・レベルは必要ありません SCLM • APAR OA27379 (SCLM 検索) PTF UA46330 + UA46331、UA46332、UA46333、UA46334 (言語依存) LE (PL/I) PTF またはサービス・レベルは必要ありません TN3270 PTF またはサービス・レベルは必要ありません
5694-A01	z/OS v 1.8	HLASM PTF またはサービス・レベルは必要ありません XL C/C++ PTF またはサービス・レベルは必要ありません SCLM • APAR OA21104 (通知ビルド・モード) PTF UA35046 + UA35056、UA35057、UA35058、 または UA35059 (言語に依存) • APAR OA16924 (拡張 SCLMINFO) PTF UA29772 + UA29922、UA29923、UA29924、 または UA29925 (言語に依存) • APAR OA16804 (代理ユーザー ID サポートを追加) PTF UA33524 + UA33533、UA33534、UA33535、 または UA33536 (言語に依存) LE (PL/I) • APAR PK41552 (Developer for System z 用の新しい PL/I メッセージ) PTF UK24482 (英語) または UK24483 (日本語) TN3270 PTF またはサービス・レベルは必要ありません

関連する製品 Web サイトは、次のとおりです。

<http://www-03.ibm.com/systems/z/os/zos/>

注:

1. ジョブ・モニターでアクティブなジョブの出力を表示したいと考えている JES3 ユーザーには、JES3 v 1.10 以上が相互前提条件となります。
2. Developer for System z 内で開発または編集されたアセンブラ・プログラムをコンパイルするためには、リストされているサービス・レベルが適用されたホストに高水準アセンブラ (HLASM) がインストールされている必要があります。

関連する製品 Web サイトは、次のとおりです。

<http://www.ibm.com/software/awdtools/hlasm/>

3. Developer for System z 内で開発または編集された C/C++ プログラムをコンパイルするためには、リストされているサービス・レベルが適用されたホストに XL C/C++ コンパイラーがインストールされている必要があります。

関連する製品 Web サイトは、次のとおりです。

<http://www.ibm.com/software/awdtools/czos/>

4. SCLM Developer Toolkit をサポートするためには、リストされているサービス・レベルが適用されたホストに SCLM がインストールされている必要があります。

関連する製品 Web サイトは、次のとおりです。

<http://www.ibm.com/software/awdtools/sclmsuite/sclm/>

注:

- APAR OA16804 は、セキュア・ビルド、プロモート、およびデプロイを使用する場合にのみ必要です。
 - APAR OA26997 は、メンバー・セキュリティのサポートにのみ必要です。
 - APAR OA27379 は、メンバー・セキュリティのサポートまたは SCLM 検索機能にのみ必要です。
5. PL/I 用のエンタープライズ・サービス・ツールをサポートするためには、リストされているサービス・レベルが適用されたホストに言語環境プログラムがインストールされている必要があります。

関連する製品 Web サイトは、次のとおりです。

<http://www-03.ibm.com/servers/eserver/zseries/zos/le/>

6. Host Connect Emulator をサポートするためには、リストされているサービス・レベルが適用されたホストに TN3270 がインストールされている必要があります。TN3270 は、IBM Communications Server の IP サービス・コンポーネントの一部です。

関連する製品 Web サイトは、次のとおりです。

<http://www-01.ibm.com/software/network/commserver/zos/>

COBOL コンパイラー

Developer for System z 内で開発または編集されている COBOL プログラムをコンパイルするには、以下のいずれかのレベルがホストにインストールされている必要があります。

プログラム番号	プロダクト名	必要な PTF またはサービス・レベル
5655-S71	IBM Enterprise COBOL for z/OS v 4.2	PTF またはサービス・レベルは必要ありません
5655-S71	IBM Enterprise COBOL for z/OS v 4.1	PTF またはサービス・レベルは必要ありません
5535-G53	IBM Enterprise COBOL for z/OS v 3.4	PTF またはサービス・レベルは必要ありません

関連する製品 Web サイトは、次のとおりです。

<http://www.ibm.com/software/awdtools/cobol/zos/>

注: エンタープライズ・サービス・ツールで、COBOL v 4.1 の XMLSS ベースの XML PARSE 機能を使用するコンパイル済み XML 変換を生成するには、IBM Enterprise COBOL for z/OS v 4.1 が必要です。

PL/I コンパイラー

Developer for System z 内で開発または編集されている PL/I プログラムをコンパイルするには、以下のいずれかのレベルがホストにインストールされている必要があります。

プログラム番号	プロダクト名	必要な PTF またはサービス・レベル
5655-H31	IBM Enterprise PL/I for z/OS v 3.9	PTF またはサービス・レベルは必要ありません
5655-H31	IBM Enterprise PL/I for z/OS v 3.8	PTF またはサービス・レベルは必要ありません
5655-H31	IBM Enterprise PL/I for z/OS v 3.7	PTF またはサービス・レベルは必要ありません
5655-H31	IBM Enterprise PL/I for z/OS v 3.6	PTF またはサービス・レベルは必要ありません

関連する製品 Web サイトは、次のとおりです。

<http://www.ibm.com/software/awdtools/pli/plizos/>

Debug Tool for z/OS

Developer for System z からのリモート・デバッグをサポートするには、以下のいずれかのレベルがホストにインストールされている必要があります。

プログラム番号	プロダクト名	プログラム言語	必要な APAR、PTF、またはサービス・レベル
5655-V50	IBM Debug Tool for z/OS V10.1	COBOL、PL/I、C/C++、アセンブラー、および追加フィーチャー	すべての使用可能な保守
5655-U27	IBM Debug Tool for z/OS V9.1	COBOL、PL/I、C/C++、アセンブラー、および追加フィーチャー	すべての使用可能な保守
5655-S16	IBM Debug Tool Utilities and Advanced Functions for z/OS V8.1.0	COBOL、PL/I、C/C++、アセンブラー、および追加フィーチャー	すべての使用可能な保守
5655-S17	IBM Debug Tool for z/OS V8.1.0	COBOL、PL/I、アセンブラー、C/C++	すべての使用可能な保守

注: IBM Rational Developer for System z v7.6.1 以上で CICS デバッグ構成をサポートするには、IBM Debug Tool v10.1 または v9.1 (PTF 番号 - UK52904) が必要です。

関連する製品 Web サイトは、次のとおりです。

<http://www.ibm.com/software/awdtools/debugtool/>

注: Debug Tool Utilities and Advanced Functions (DTU&AF) は、Debug Tool のスーパーセットです。

バージョン 9 で、Debug Tool for z/OS と Debug Tool Utilities and Advanced Functions が統合されて 1 つのオファリングになりました。

CICS Transaction Server

CICS ステートメントが埋め込まれたアプリケーションをサポートするには、以下のいずれかのレベルがインストールされている必要があります。

プログラム番号	プロダクト名	必要な PTF またはサービス・レベル
5655-S97	IBM CICS Transaction Server for z/OS v 4.1	PTF またはサービス・レベルは必要ありません
5697-E93	IBM CICS Transaction Server for z/OS v 3.2	UK34221
5697-E93	IBM CICS Transaction Server for z/OS v 3.1	UK15767、UK15764、UK11782、UK11294、UK12233、UK12521、UK15261、UK15271、UK34221、UK34078

関連する製品 Web サイトは、次のとおりです。

<http://www.ibm.com/software/hp/cics/tserver/>

注:

- CICS Transaction Server をデバッグ・ツールと連動させるには、追加の構成が必要です。
- IBM Rational Developer for System z v 7.6 以上の新機能である Application Deployment Manager、Service Component Architecture、およびエンタープライズ・サービス・ツールの各フィーチャーをサポートするには、CICS Transaction Server v 4.1 以上で提供されている RESTful インターフェースが必要です。
- エンタープライズ・サービス・ツールの多くのフィーチャーをサポートするには、CICS Transaction Server v 3.2 以上が必要です。

ランタイム要件の特性の完全なリストについては、IBM Rational Developer for System z インフォメーション・センター (<http://publib.boulder.ibm.com/infocenter/ratdevz/v7r6/>) にあるエンタープライズ・サービス・ツールの資料を参照してください。

- Application Deployment Manager には、最小でもサービス UK34221 を備えた CICS Transaction Server v 3.1 が必要です。

IMS

IMS データベースおよびデータ通信を使用するアプリケーションをサポートするには、以下のいずれかのレベルがホストにインストールされている必要があります。

プログラム番号	プロダクト名	必要な PTF またはサービス・レベル
5635-A02	IBM IMS v 11.1	PTF またはサービス・レベルは必要ありません
5635-A01	IBM IMS v 10.1	PTF またはサービス・レベルは必要ありません
5655-J38	IBM IMS v 9.1	PTF またはサービス・レベルは必要ありません

関連する製品 Web サイトは、次のとおりです。

<http://www.ibm.com/software/data/ims/ims/>

注:

- IMS をデバッグ・ツールと連動させるには、追加の構成が必要です。
- エンタープライズ・サービス・ツールには、IMS、IMS Connect、および IMS SOAP ゲートウェイのバージョン 10.1 以上が必要です。

DB2 for z/OS

DB2 をサポートするには、以下のいずれかのレベルがホストにインストールされている必要があります。

プログラム番号	プロダクト名	必要な PTF またはサービス・レベル
5635-DB2	IBM DB2 for z/OS v 9.1	PTF またはサービス・レベルは必要ありません
5625-DB2	IBM DB2 Universal Database™ for z/OS v 8.1	PTF またはサービス・レベルは必要ありません

関連する製品 Web サイトは、次のとおりです。

<http://www.ibm.com/software/data/db2/zos/>

Rational Team Concert for System z

Developer for System z のリモート・プロジェクトを使用した Jazz ベースのソース制御には、以下のレベルがインストールされている必要があります。

プログラム番号	プロダクト名	必要な PTF またはサービス・レベル
5724-V82	Rational Team Concert for System z Server v 2.0	<p>FMID HAHA200 - Team Server (チーム・サーバー)</p> <ul style="list-style-type: none"> • UK54064 • UK54071 • UK54073 • UK54095 • UK54098 <p>FMID HAHB200 - Toolkit (ツールキット)</p> <ul style="list-style-type: none"> • UK54065 • UK54066 • UK54099 <p>FMID HAHC200 - Job Monitor (ジョブ・モニター)</p> <ul style="list-style-type: none"> • PTF またはサービス・レベルは必要ありません <p>FMID HAHD200 - BuildForge Agent (BuildForge エージェント)</p> <ul style="list-style-type: none"> • UK54097

関連する製品 Web サイトは、次のとおりです。

<http://www-01.ibm.com/software/awdtools/rtc/>

注: Rational Team Concert Server v 1.0 または Rational Team Concert for System z Server v 1.0.1 は、一部の Developer for System z 機能 (ローカル・プロジェクトなど) を選択的にサポートします。統合性を高め、すべての機能に対応できるようにするには、Rational Team Concert for System z Server v 2.0.0.2 をお勧めします。

File Manager

File Manager Integration をサポートするには、以下のいずれかのレベルがホストにインストールされている必要があります。

プログラム番号	プロダクト名	必要な PTF またはサービス・レベル
5655-U29	IBM File Manager for z/OS v 10.1	• UK54428

関連する製品 Web サイトは、次のとおりです。

<http://www.ibm.com/software/awdtools/filemanager/>

Fault Analyzer

Fault Analyzer Integration をサポートするには、以下のレベルがホストにインストールされている必要があります。

プログラム番号	プロダクト名	必要な PTF またはサービス・レベル
5655-V51	IBM Fault Analyzer v 10.1	PTF またはサービス・レベルは必要ありません
5655-U28	IBM Fault Analyzer v 9.1	PTF またはサービス・レベルは必要ありません
5655-S15	IBM Fault Analyzer v 8.1	PTF またはサービス・レベルは必要ありません

関連する製品 Web サイトは、次のとおりです。

<http://www.ibm.com/software/awdtools/faultanalyzer/>

REXX

SCLM Developer Toolkit をサポートするには、以下のいずれかのレベルがホストにインストールされている必要があります。

プログラム番号	プロダクト名	必要な PTF またはサービス・レベル
5695-014	zSeries v 1.4 での IBM Library for REXX	PTF またはサービス・レベルは必要ありません
5695-014	zSeries Alternate Library v 1.4.0 (FMID HWJ9143、JWJ9144) での IBM Library for REXX	PTF またはサービス・レベルは必要ありません

REXX/370 Alternate Library のバージョンは、製品 Web サイトから入手できます。

<http://www.ibm.com/software/awdtools/rexx/rexxzseries/>

Ported Tools

SCLM Developer Toolkit で sftp または scp を使用してセキュア・デプロイメントを行うには、IBM Ported Tools for z/OS が (z/OS UNIX に) インストールされている必要があります。

IBM Ported Tools for z/OS のバージョンは、製品 Web サイトから入手できます。

http://www-03.ibm.com/servers/eserver/zseries/zos/unix/port_tools.html

Ant

SCLM Developer Toolkit で JAVA/J2EE ビルドを行うには、Apache Ant が (z/OS UNIX に) インストールされている必要があります。

Apache Ant は Java をベースにしたオープン・ソースのビルド・ツールで、製品 Web サイトからダウンロードできます。

<http://ant.apache.org/>

Endevor®

Developer for System z Interface for CA Endevor® SCM を使用するには、CA Endevor® Software Change Manager リリース 12 をインストールする必要があります。

CA Endevor® SCM は、CA の製品です。関連する製品 Web サイトは、次のとおりです。

<http://www.ca.com/us/products/product.aspx?ID=259>

参考文献

参考資料

本書では、以下の資料を参照しています。

表 52. 参考資料

資料名	資料番号	参照	参照 Web サイト
Java Diagnostic Guide	SC34-6650	Java 5.0	http://www.ibm.com/developerworks/java/jdk/diagnosis/
Java SDK and Runtime Environment User Guide	/	Java 5.0	http://www-03.ibm.com/servers/eserver/zseries/software/java/
IBM Rational Developer for System z Program Directory	GI88-4172	Developer for System z	http://www-306.ibm.com/software/awdtools/rdz/library/
Rational Developer for System z Common Access Repository Manager Developer's Guide	SC23-7660	Developer for System z	http://www-306.ibm.com/software/awdtools/rdz/library/
Rational Developer for System z 前提条件	SC88-4704	Developer for System z	http://www-306.ibm.com/software/awdtools/rdz/library/
Rational Developer for System z ホスト構成クイック・スタート・ガイド	GI88-4171	Developer for System z	http://www-306.ibm.com/software/awdtools/rdz/library/
Rational Developer for System z ホスト・プランニング・ガイド	GI88-4131	Developer for System z	http://www-306.ibm.com/software/awdtools/rdz/library/
SCLM Developer Toolkit 管理者ガイド	SC88-5664	Developer for System z	http://www-306.ibm.com/software/awdtools/rdz/library/
APPC and WebSphere Developer for System z	SC23-5885	ホワイト・ペーパー	http://www-306.ibm.com/software/awdtools/rdz/library/
Communications Server IP 構成ガイド	SC88-8926	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Communications Server IP 構成解説書	SC88-8927	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Communications Server IP Diagnosis Guide	GC31-8782	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Communications Server IP システム管理者のコマンド	SC88-9073	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Communications Server SNA ネットワーク・インプリメンテーション・ガイド	SC88-8928	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Communications Server SNA オペレーション	SC88-8930	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/

表 52. 参考資料 (続き)

資料名	資料番号	参照	参照 Web サイト
Cryptographic Services System SSL (Secure Sockets Layer) プログラミング	SD88-6252	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
DFSMS Macro Instructions for Data Sets	SC26-7408	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
DFSMS データ・セットの使用法	SC88-9114	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
言語環境プログラム カスタマイズ	SA88-8552	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
言語環境プログラム デバッグ・ガイド	GA88-8548	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
MVS 初期設定およびチューニング ガイド	SA88-8563	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
MVS 初期設定およびチューニング解説書	SA88-8564	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
MVS JCL 解説書	SA88-8569	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
MVS 計画 APPC/MVS 管理	SA88-8571	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
MVS 計画: ワークロード管理	SA88-8574	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
MVS システム・コマンド	SA88-8593	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Security Server RACF コマンド言語解説書	SA88-8617	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Security Server RACF セキュリティー管理者のガイド	SA88-8613	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
TSO/E カスタマイズ	SA88-8629	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
TSO/E REXX 解説書	SA88-8635	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
UNIX System Services コマンド解説書	SA88-8641	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
UNIX System Services 計画	GA88-8639	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
UNIX システム・サービス ユーザーズ・ガイド	SA88-8640	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
REXX および z/OS UNIX システム・サービスの使い方	SA88-8644	z/OS 1.9	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
Resource Definition Guide	SC34-6430	CICSTS 3.1	http://www-03.ibm.com/systems/z/os/zos/bkserv/zapplsbooks.html
Resource Definition Guide	SC34-6815	CICSTS 3.2	http://www-03.ibm.com/systems/z/os/zos/bkserv/zapplsbooks.html
Resource Definition Guide	SC34-7000	CICSTS 4.1	https://publib.boulder.ibm.com/infocenter/cicsts/v4r1/index.jsp?topic=/com.ibm.cics.ts.home.doc/library/library_html.html
RACF Security Guide	SC34-6454	CICSTS 3.1	http://www-03.ibm.com/systems/z/os/zos/bkserv/zapplsbooks.html

表 52. 参考資料 (続き)

資料名	資料番号	参照	参照 Web サイト
RACF Security Guide	SC34-6835	CICSTS 3.2	http://www-03.ibm.com/systems/z/os/zos/bkserv/zapplsbooks.html
RACF Security Guide	SC34-7003	CICSTS 4.1	https://publib.boulder.ibm.com/infocenter/cicsts/v4r1/index.jsp?topic=/com.ibm.cics.ts.home.doc/library/library_html.html
言語解説書	SC88-9117	Enterprise COBOL for z/OS	http://www-03.ibm.com/systems/z/os/zos/bkserv/zapplsbooks.html

本書では、以下の Web サイトを参照しています。

表 53. 参照される Web サイト

説明	参照 Web サイト
Developer for System z インフォメーション・センター	http://publib.boulder.ibm.com/infocenter/ratdevz/v7r6/index.jsp
Developer for System z サポート	http://www-306.ibm.com/software/awdtools/rdz/support/
Developer for System z ライブラリー	http://www-306.ibm.com/software/awdtools/rdz/library/
Developer for System z ホーム・ページ	http://www-306.ibm.com/software/awdtools/rdz/
Developer for System z 推奨サービス	http://www-01.ibm.com/support/docview.wss?rs=2294&context=SS2QJ2&uid=swg27006335
Developer for System z 機能拡張要求	https://www.ibm.com/developerworks/support/rational/rfe/
z/OS インターネット・ライブラリー	http://www-03.ibm.com/servers/eserver/zseries/zos/bkserv/
CICSTS インフォメーション・センター	https://publib.boulder.ibm.com/infocenter/cicsts/v4r1/index.jsp
Apache Ant のダウンロード	http://ant.apache.org/
Java keytool の資料	http://java.sun.com/j2se/1.5.0/docs/tooltools/solaris/keytool.html
CA サポート・ホーム・ページ	https://support.ca.com/

情報資料

以下の資料は、必要なホスト・コンポーネントのセットアップの問題を理解するのに役立ちます。

表 54. 情報資料

資料名	資料番号	参照	参照 Web サイト
ABCs of z/OS System Programming Volume 9 (z/OS UNIX)	SG24-6989	Redbook	http://www.redbooks.ibm.com/
System Programmer's Guide to: Workload Manager	SG24-6472	Redbook	http://www.redbooks.ibm.com/
TCPIP Implementation Volume 1: Base Functions, Connectivity, and Routing	SG24-7532	Redbook	http://www.redbooks.ibm.com/

表 54. 情報資料 (続き)

資料名	資料番号	参照	参照 Web サイト
TCP/IP Implementation Volume 3: High Availability, Scalability, and Performance	SG24-7534	Redbook	http://www.redbooks.ibm.com/
TCP/IP Implementation Volume 4: Security and Policy-Based Networking	SG24-7535	Redbook	http://www.redbooks.ibm.com/

用語集

[ア行]

アイソモアフィック (Isomorphic). ルートから始まる XML インスタンス文書の構成された各エレメント (つまり、他のエレメントを含んでいるエレメント) は、唯一の対応する COBOL グループ項目を持ち、そのグループ項目のネストの深さは、同等の XML 項目のネストの深さと同一である。トップから始まる XML インスタンス文書の構成されていない各エレメント (つまり、他のエレメントを含んでいないエレメント) は、唯一の対応する COBOL 基本項目を持っており、その基本項目のネストの深さは、同等の XML 項目のネスト・レベルと同じであり、実行時のメモリー・アドレスは、一意に識別できる。

アクション ID (Action ID). アクションを表す 0 から 999 までの数値 ID。

アプリケーション・サーバー (Application Server).

1. ブラウザー・ベースのコンピューターと組織のバックエンド・ビジネス・アプリケーションまたはデータベースとの間で、すべてのアプリケーション操作を処理するプログラム。J2EE 規格に準拠した、Java ベースの特殊クラスの appserver が存在する。これらの appserver 間では J2EE コードを簡単に移植できる。動的 Web コンテンツ用の JSP とサーブレット、およびトランザクションとデータベース・アクセス用の EJB をサポートできる。
2. リモート・アプリケーションからの要求のターゲット。DB2 環境において、アプリケーション・サーバー機能は分散データ機能によって提供され、リモート・アプリケーションの DB2 データにアクセスするために使用される。
3. アプリケーション・プログラムの実行環境を提供する分散ネットワーク内のサーバー・プログラム。
4. アプリケーション・リクエストからの要求のターゲット。アプリケーション・サーバー・サイトのデータベース管理システム (DBMS) は、要求されたデータを提供する。
5. Content Manager のアセットおよび照会を要求するクライアントとの通信を処理するソフトウェア。

インタープリター (Interpreter). 高水準プログラミング言語の 1 つの命令を翻訳および実行してから、次の命令を翻訳および実行するプログラム。

エラー・バッファー (Error Buffer). エラー出力情報を一時的に保持するために使用されるストレージの部分。

応答ファイル (Response File).

1. プログラムからの質問に対する定義済みの回答セットが入っているファイル。それらの値を一度に 1 つずつ入力する代わりに使用される。
2. インストールを自動化するセットアップおよび構成データを使用してカスタマイズできる ASCII ファイル。セットアップおよび構成データは本来、対話式インストールのときに入力すべきものだが、応答ファイルを使用すると、インストールを介入なしに進行させることができる。

[カ行]

ゲートウェイ (Gateway).

1. Web サービス呼び出しのとき、インターネットとイントラネット環境のブリッジとなるミドルウェア・コンポーネント。
2. エンドポイントとそれ以外の Tivoli® 環境の間でサービスを提供するソフトウェア。
3. Voice over Internet Protocol 環境と回路交換環境の間のブリッジとなる VoIP のコンポーネント。
4. 異なるネットワーク・アーキテクチャーを備えたネットワーク、またはシステムを接続するために使用されるデバイスまたはプログラム。各システムが異なる特性 (例えば、異なる通信プロトコル、異なるネットワーク・アーキテクチャー、異なるセキュリティ・ポリシーなど) を備えている場合があり、ゲートウェイは、そのような場合に変換の役割と接続の役割を果たす。

コンテナ (Container).

1. CoOperative Development Environment/400 においては、ソース・ファイルを格納および編成するシステム・オブジェクト。コンテナの例としては、i5/OS® ライブラリーや MVS 区分データ・セットがある。
2. J2EE において、コンポーネントにライフ・サイクル管理、セキュリティ、デプロイメント、およびランタイム・サービスを提供するエンティティ。(Sun) 各タイプのコンテナ (EJB、Web、JSP、サーブレット、アプレット、およびアプリケーション・クライアント) はコンポーネント固有のサービスも提供する。

3. バックアップ・リカバリーおよびメディア・サービスにおいて、ボックス、ケース、またはラックなどのメディアの保管と移動に使用される物理オブジェクト。
4. 仮想テープ・サーバー (VTS) において、エクスポートされた 1 つ以上の論理ボリューム (LVOL) を保管できる貯蔵所。1 つ以上の LVOL を含み、VTS ライブラリーの外部に存在するボリューム・スタックは、そのボリュームのコンテナと見なされる。
5. データの物理的な記憶場所。例えば、ファイル、ディレクトリ、デバイスなど。
6. ポートレットまたはページ上にあるその他のコンテナのレイアウトを調整するために使用される列または行。
7. オブジェクトを保持するユーザー・インターフェースの要素。フォルダー・マネージャーにおいて、他のフォルダーまたは文書を格納できるオブジェクト。

コンパイル (Compile).

1. 統合化言語環境 (Integrated Language Environment) (ILE) 言語において、ソース・ステートメントを、プログラムまたはサービス・プログラムにバインドできるモジュールに変換すること。
2. 高水準言語で表現されたプログラムの全部または一部を、中間言語、アセンブリ言語、またはマシン言語で表現されたコンピューター・プログラムに変換すること。

[サ行]

サーバー・ビュー (Servers View). 使用しているすべてのサーバーとそれに関連した構成をリストとして表示する。

サイレント・アンインストール (Silent Uninstallation). アンインストール・コマンドが起動された後、コンソールにメッセージを送らず、その代わりにメッセージとエラーをログ・ファイルに保管するアンインストール・プロセス。

サイレント・インストール (Silent Installation). コンソールにメッセージを送らず、その代わりにメッセージとエラーをログ・ファイルに保管するインストール。また、サイレント・インストールはデータ入力に応答ファイルを使用できる。

シェル (Shell). ユーザーとオペレーティング・システム間のソフトウェア・インターフェース。コマンドおよびユーザー対話を解釈し、ユーザーとオペレーティング・システム間の通信を行う。さまざまなレベルのユ

ーザー対話を処理するために、1 つのコンピューターに、複数のレイヤーのシェルが存在する場合がある。

シェル名 (Shell Name). シェル・インターフェースの名前。

シェル・スクリプト (Shell Script). シェルが解釈できる、コマンドが入ったファイル。ユーザーは、シェル・コマンド・プロンプトでスクリプト・ファイルの名前を入力して、シェルにスクリプト・コマンドを実行させる。

出力ビュー (Output View). 処理対象のオブジェクトに関連したメッセージ、パラメーター、および結果を表示する。

双方向 (Bidirectional) (bi-di). 数字 (左から右に書かれる) を除き、一般に右から左に書かれるアラビア語やヘブライ語などのスクリプトに関する用語。この定義は、Localization Industry Standards Association (LISA) の用語集からのものである。

双方向属性 (Bidirectional Attribute). テキスト・タイプ、テキスト方向、数値スワッピング、および対称スワッピング。

[タ行]

対話式システム生産性向上機能 (Interactive System Productivity Facility) (ISPF). フルスクリーン・エディターおよびダイアログ・マネージャーとして機能する IBM ライセンス・プログラム。アプリケーション・プログラムの作成に使用され、標準的な画面パネルとアプリケーション・プログラマーと端末ユーザーとの間に対話式ダイアログを生成する手段となる。ISPF は、DM、PDF、SCLM、および C/S という 4 つの主要なコンポーネントからなる。DM コンポーネントとはダイアログ・マネージャーのことで、ダイアログとエンド・ユーザーにサービスを提供する。PDF コンポーネントとはプログラム開発機能のことで、ダイアログまたはアプリケーション開発者を支援するサービスを提供する。SCLM コンポーネントとは Software Configuration Library Manager のことで、アプリケーション開発者にアプリケーション開発ライブラリーを管理するためのサービスを提供する。C/S コンポーネントとは、クライアント/サーバーのことで、これによってプログラマブル・ワークステーション上で ISPF を実行し、ワークステーションのオペレーティング・システムの表示機能を使用してパネルを表示し、ワークステーションのツールおよびデータをホストのツールおよびデータと統合することができる。

タスク・リスト (Task List). 単一の制御フローによって実行できる手順のリスト。

データ定義ビュー (Data Definition View). データベースとそのオブジェクトのローカル表現が入っており、それらのオブジェクトを操作してリモート・データベースにエクスポートする機能を提供する。

データベース (Database). 1 つ以上のアプリケーションに対してサービスを行うために、まとめて保管される、相互に関連するか独立したデータ項目のコレクション。

データ・セット (Data Set). データの保管と取り出しの主要単位。いくつかの規定された配置の 1 つに置かれたデータのコレクションで構成され、システムがアクセスする制御情報によって記述される。

デバッグ (Debug). プログラム内のエラーを検出、診断、および除去すること。

デバッグ・セッション (Debugging Session). 開発者がデバッガーを開始した時点から、デバッガーを終了する時点までに発生するデバッグ・アクティビティー。

【ナ行】

ナビゲーター・ビュー (Navigator View). ワークベンチ内のリソースの階層図を提供する。

【ハ行】

パースペクティブ (Perspective). ワークベンチ内のリソースのさまざまな側面を表示するビューのグループ。ワークベンチ・ユーザーは、手元のタスクに応じてパースペクティブを切り替え、パースペクティブ内のビューおよびエディターのレイアウトをカスタマイズできる。

非アイソモフィック (Non-Isomorphic). 形態が同一でない (非アイソモフィック) XML 文書と COBOL グループに属する、COBOL 項目と XML エLEMENT との単純なマッピング。非アイソモフィック・マッピングは、アイソモフィック構造の非アイソモフィック・ELEMENT 間でも作成できる。

ビルド要求 (Build Request). ビルド・トランザクションの実行を求めるクライアントからの要求。

ビルド・トランザクション (Build Transaction). クライアントからビルド要求を受信した後に、MVS 上で開始されるジョブ。

【ラ行】

リポジトリ (Repository).

1. データのストレージ域。すべてのリポジトリは名前と、関連するビジネス項目タイプを備えている。デフォルトでは、名前はビジネス項目の名前と同じものになる。例えば、送り状のリポジトリは「Invoices」と呼ばれる。ローカル (プロセスに固有のもの) とグローバル (再利用可能なもの) の 2 つのタイプの情報リポジトリがある。
2. BTS プロセスの状態を保管している VSAM データ・セット。プロセスが BTS の制御下で実行されていない場合、そのプロセスの状態 (およびそのプロセスを構成するアクティビティーの状態) は、リポジトリ・データ・セットに書き込まれることによって保存される。特定のプロセス・タイプに属するすべてのプロセス (およびそれらのアクティビティー・インスタンス) の状態は、同じリポジトリ・データ・セットに保管される。複数のプロセス・タイプのレコードを同じリポジトリに書き込むことができる。
3. ソース・コードおよびその他のアプリケーション・リソース用の永続的なストレージ域。チーム・プログラミング環境においては、共用リポジトリによってアプリケーション・リソースへのマルチユーザー・アクセスが可能になる。
4. クラスターのメンバーであるキュー・マネージャーに関する情報のコレクション。この情報には、キュー・マネージャーの名前、ロケーション、チャンネル、ホスト対象となるキューなどが含まれる。

リポジトリ・インスタンス (Repository Instance). SCM 内に存在するプロジェクトまたはコンポーネント。

リポジトリ・ビュー (Repositories View). ワークベンチに追加された CVS リポジトリ・ロケーションを表示する。

リモート・システム (Remote System). ネットワーク内にあり、使用しているシステムの通信相手にできる別のシステム。

リモート・ファイル・システム (Remote File System). 別のサーバーまたはオペレーティング・システム上にあるファイル・システム。

リンケージ・セクション (Linkage Section). アクティブにされる単位 (呼び出されたプログラムまたはメソッド) のデータ部の中に含まれており、アクティブにする単位 (プログラムまたはメソッド) から使用可能なデー

タ項目を記述しているセクション。これらのデータ項目は、アクティブにされる単位とアクティブにする単位の両方から参照できる。

ロード・ライブラリー (Load Library). ロード・モジュールを含んでいるライブラリー。

ロック・アクション (Lock Action). メンバーをロックする。

[特殊文字と記号]

「出力コンソール」ビュー (Output Console View). プロセスの出力を表示し、プロセスへのキーボード入力を提供する。

「リモート・システム」パースペクティブ (Remote Systems Perspective). ISPF に似た規約を使用して、リモート・システムを管理するインターフェースを提供する。

R

RAM. Repository Access Manager の略。

S

SIDEDECK. DLL プログラムの機能を公開するライブラリー。ソース・コードがコンパイルされた後、入り口名とモジュール名は、このライブラリーに保管される。

U

URL. Uniform Resource Locator の略。

IBM Rational Developer for System z 資料に関する特記事項

実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒242-8502
神奈川県大和市下鶴間1623番14号
日本アイ・ビー・エム株式会社
法務・知的財産
知的財産権ライセンス渉外

以下の保証は、国または地域の法律に沿わない場合は、適用されません。IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態を提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

Intellectual Property Dept. for Rational Software
IBM Corporation
3039 Cornwallis Road, PO Box 12195
Research Triangle Park, NC 27709
U.S.A.

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性があります。その測定値が、一般に利用可能なシステムのもと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確認できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者をお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があります、単に目標を示しているものです。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほのめかしたり、保証することはできません。サンプル・プログラムは、現存するままの状態を提供され、いかなる保証条件も適用されません。IBM は、これらのサンプル・プログラムの使用から生ずるいかなる損害に対しても責任を負いません。

商標

IBM、IBM ロゴ、および ibm.com は、世界の多くの国で登録された International Business Machines Corp. の商標または登録商標です。他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。現時点での IBM の商標リストについては、www.ibm.com/legal/copytrade.shtml をご覧ください。

Intel® および Pentium® は、Intel Corporation または子会社の米国およびその他の国における商標または登録商標です。

Microsoft®、Windows®、および Windows ロゴは、Microsoft Corporation の米国およびその他の国における商標です。

Java およびすべての Java 関連の商標およびロゴは、Sun Microsystems, Inc. の米国およびその他の国における商標です。

UNIX は The Open Group の米国およびその他の国における登録商標です。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

【ア行】

アクション、ジョブに対する - 実行の制限 179
アクセス、スプール・ファイルへの条件付き 180
アクセスの改善、システム・ライブラリーへの 262
アクセスの要件 10
アクセス方式、TSO 283
アクセス方式の使用、TSO/ISPF クライアント・ゲートウェイ 284
アクティブ化、共通アクセス・リポジトリ・マネージャの 55
アクティブ化、サンプルの Repository Access Manager (RAM) の 67
アクティブ化、スケルトン Repository Access Manager の 68
アクティブ化、CA Endeavor® Software Configuration Manager Repository Access Manager の 68
アクティブ化、PDS Repository Access Manager の 67
アクティブ化、SCLM Repository Access Manager の 68
アドレス・スペースの数 229
アドレス・スペース・サイズ 162
アドレス・スペース・サイズの限度 239
アプリケーション開発 262
アプリケーション保護の定義、RSE の 200
暗号化、SSL を使用した通信 172
暗号化、ssl.properties 100
暗号化通信、SSL 181, 187, 273
依存関係、ホスト名 329
インストール検査 117
インストール検査プログラム 120
インストールと構成、ホスト・コンポーネントの 3
インストール前の考慮事項 4
インストール・ロギング、CICS リソース 271
インターフェース、RESTful と Web サービス 81
エラー・フィードバック・トレース 156

エラー・メッセージ、IRZ 診断用 99
エンタープライズ・サービス・ツール・サポート 97
オプションのカスタマイズ・タスク 95
オプションのコンポーネント 53
オペランド、構文図での複数の選択 144
オペランド、構文図の 143
オペレーター・コマンド 133

【力行】

解決されないホスト・アドレス、TCP/IP リゾルバー
lock.log 336
開始タスク、検査 117
開始タスク、サービスの検査 120
開始タスクの定義、Developer for System z
JMON 開始タスク 196
LOCKD 開始タスク 196
RSED 開始タスク 196
改善、システム・ライブラリーへのアクセスの 262
開発、アプリケーション 262
外部通信 173
外部通信の限定、指定したポート 171
鍵ストアの作成、keytool による 327
鍵データベースの作成、gskkyman による 325
鍵リングの作成、RACF による 317
カスタマイズ - ISPF.conf 284
カスタマイズ、基本的な 17
カスタマイズ、Application Deployment Manager 269
カスタマイズ、CRD サーバー・トランザクション ID の 83
カスタマイズ、SCLM Developer Toolkit の 87
カスタマイズ、TSO 環境の 283
カスタマイズのセットアップ 18
カスタマイズ・タスク、オプションの 95
仮想記憶通信アクセス方式 350
可用性、ポート 121
環境設定、INETD 347
監査制御
audit.* options 176
daemon.log 176
enable.audit.log 176
_RSE_HOST_CODEPAGE 176
監査データ
ログに記録されるアクション 177

監査ロギング、RSE デーモンで管理 176
管理、ワークロード 263
管理ユーティリティ、マイグレーションに関する注 277
管理ユーティリティのメッセージ 278
記号、構文図の 143
既存の ISPF プロファイル (APPC トランザクション JCL) の使用 287
基本的なカスタマイズ 17
基本リゾルバー構成ファイル 331
キャッシュ管理ユーティリティ、Java 仮想マシン (JVM) 266
キャッシュ・サイズ制限、Java 仮想マシン (JVM) 265
キャッシュ・セキュリティ、Java 仮想マシン (JVM) 265
共存、共存を可能にするための
rsed.envvars の更新 319
共通アクセス・リポジトリ・マネージャ (CARMA)、FMID HCMA710 311
共通アクセス・リポジトリ・マネージャのアクティブ化 55
共通アクセス・リポジトリ・マネージャ・ロギング 150
許可ビット、z/OS UNIX 157
拒否、接続の 166
クイック・スタート、Java オプション (-Xquickstart) 264
クライアント認証、X.509 証明書を使用した 182
クライアント認証サポートの追加、X.509 324
クライアント・ゲートウェイ構成ファイル、TSO/ISPF 51
クライアント・ゲートウェイ・アクセス方式の使用、TSO/ISPF 284
クライアント・チェックリスト 14
クラス共用、Java 仮想マシン (JVM) 間 264
クラス共用、Java 仮想マシン (JVM) での有効化 265
クラス共用の有効化、Java 仮想マシン (JVM) 265
クローン作成、既存の RSE セットアップ 319
言語環境プログラム・ランタイム・ライブラリー 262
検査、セキュリティ設定 202
検査、ISPF の TSO/ISPF クライアント・ゲートウェイ接続 124

検査、REXEC/SSH シェル・スクリプト 129
検索順序、構成情報の 330
検索順序、ネイティブ MVS 環境 342
検索順序、z/OS UNIX 環境 331, 341
限度、システム 165
限度、BPXPRMxx で設定される z/OS UNIX の 19
向上、セキュリティ検査のパフォーマンスの 263
更新特権、非システム管理者 215
構成、ホスト・コンポーネントの 3
構成可能なファイル 303, 312
構成情報の検索順序 330
構成ファイル、基本リゾルパー 331
構成ファイル、同一のソフトウェア・レベルの、異なる 292
構成ファイル、Developer for System z 188
構成ファイルの変更、他の製品 298
構成前の考慮事項 9
構成問題のトラブルシューティング 145
構文図、複数のオペランドの選択 144
構文図、1 行より長い 144
構文図のオペランド 143
構文図の記号 143
構文図の非英数字およびブランク・スペース 144
構文図の読み方 143
構文フラグメント 144
構文例 144
考慮事項、インストール前の 4
考慮事項、構成前の 9
考慮事項、サーバーに関する 11
考慮事項、セキュリティに関する 169
考慮事項、デプロイメント前の 13
考慮事項、パフォーマンスに関する 261
考慮事項、ユーザー ID に関する 10
固定 Java ヒープ・サイズ 264
異なる構成ファイル、同一のソフトウェア・レベル 292
コマンド、オペレーター 133
コマンド、ロック・デーモン Modify 139
コマンド、ロック・デーモン Start 134
コマンド、JES ジョブ・モニター Modify 135
コマンド、JES ジョブ・モニター Start 133
コマンド、Modify (F) 135
コマンド、RSE デーモン Modify 136
コマンド、RSE デーモン Start 134
コマンド、Start (S) 133
コマンド、Stop (P) 140
コマンド・セキュリティの定義、JES 197
コンソール・メッセージ 140

コンソール・メッセージ (続き)
ロック・デーモン 140
JES ジョブ・モニター 140
RSE スレッド・プール・サーバー 140
RSE デーモン 140
コンポーネント、オプションの 53
コンポーネントのインストールと構成 3
コンポーネントの概要、Developer for System z
グラフィカル表現 205

[サ行]

サード・パーティーおよび X.509 証明書 170
サーバーに関する考慮事項 11
サービス接続、TSO コマンド (APPC) 126
サイズ、アドレス・スペース 162
サイズの限度、アドレス・スペース 239
サイズの限度、Java ヒープ 239
サイズ見積もり、ガイドライン 240
作成、LSTRANS.FILE 90
サブシステム・タイプ
ASCH 218
CICS 218
JES 218
OMVS 218
STC 218
サポート、エンタープライズ・サービス・ツール 97
サポート、EST 97
サポートされるアプリケーション 4
サポートされるオペレーティング・システム 3
サポートされるサブシステム 4
サポートされるプラットフォーム 3
サポートの定義、RSE の PassTicket 200
さまざまなリソース定義 250
EXEC カード、サーバー JCL 250
FEJJCNGF 251
SYS1.PARMLIB(ASCHPMxx) 252
SYS1.PARMLIB(IEASYSxx) 251
SYS1.PARMLIB(IVTPRMxx) 251
参考資料 369
サンプルのストレージ、使用量分析 240
サンプル・セットアップ 256
限度の定義 257
最小限度の特定 257
スレッド・プールの数 256
シェル・スクリプト、REXEC/SSH 129
シェル・セッション、INETD の始動 345
システム限度 165
システム出口によって強制される制限 163

システム・ライブラリーへのアクセスの改善 262
シスプレックス、同一セットアップ 291
実行、複数のインスタンスの 291
実行の制限、ジョブに対するアクション 179
失敗、MVS データ・セットのオープン 166
指定したポートとの外部通信、限定 171
始動 JCL の要件 162
始動、代替の CARMA サーバーの 65
始動、z/OS UNIX INETD 343
始動パラメーターの定義、_RSE_CMDSERV_OPTS での追加 Java 51
始動パラメーターの定義、_RSE_JAVAOPTS での追加 Java 46
修飾子チェックリスト、ELAXF* 29
主接続領域、CICS 82
主接続領域と非主接続領域 270
主要なリソース定義 247
rsed.envvars 247
SYS1.PARMLIB(BPXPRMxx) 248
使用、既存の ISPF プロファイルの 284
使用、割り振り exec の 285
使用、PassTicket 175
照会、証明書失効リスト (CRL)
CRL 環境変数 184
rsed.envvars 184
条件付きアクション、ジョブに対する 177
条件付きアクセス、スプール・ファイルへの 180
使用に関する考慮事項、APPC 114
使用の回避、STEPLIB の 261
証明書、X.509 170
証明書、X.509 を使用したクライアント認証 182
証明書失効リスト (CRL) の照会
CRL 環境変数 184
rsed.envvars 184
使用量分析、サンプルのストレージ 240
初期化、IVP 120
ジョブに対する条件付きアクション 177
ジョブ・モニター・サーバー、JES 24
署名付き証明書、自己署名または認証局による署名 318
資料、参考 369
診断用 IRZ エラー・メッセージ 99
スケルトン Repository Access Manager のアクティブ化 68
スティッキー・ビット、z/OS UNIX で MVS ロード・モジュールを使用可能にする 160
ストアード・プロシージャ、DB2 95
ストレージの使用量 239

スプール・ファイルへの条件付きアクセス 180
スペース、構文図 144
スペースの使用量、z/OS UNIX ファイル・システム 244
スレッドの数 235
スレッド・プール、ロギング 148
制御ライブラリーの定義、RSE の MVS 199
製品、前提条件の 5
製品およびソフトウェアの構成、必要な 10
セキュアな APPC セットアップ、VTAM 355
セキュアな z/OS UNIX サーバーとしての RSE の定義 199
セキュア・シェル、使用 109
セキュリティ、接続 171
セキュリティ、トランザクション 272
セキュリティ、パイプライン 271
セキュリティ、リソース 273
セキュリティ、Application Deployment Manager (ADM) 271
セキュリティ、CICSTS 187
セキュリティ、INETD 345
セキュリティ、JES 177
セキュリティ、SCLM 187
セキュリティ検査のパフォーマンスの向上 263
セキュリティ設定、検査 202
セキュリティ定義 29, 189
セキュリティ定義、チェックリスト 190
セキュリティに関する考慮事項 169
セキュリティの設定およびクラスのアクティブ化 191
セキュリティの定義、JES コマンド 197
セキュリティ・ソフトウェアによる認証 184
セキュリティ・プロファイル内に保管される制限 163
セグメントの定義、OMVS 192
接続、ロック・デーモン 124
接続、JES ジョブ・モニター 123
接続、REXEC 128
接続、RSE デーモン 123
接続セキュリティ 171
接続の拒否 166
接続のフロー 210
 グラフィカル表現 210
接続領域、主と非主 270
接続領域、CICS 非主 82
設定およびクラスのアクティブ化、セキュリティの 191
セットアップ、カスタマイズの 18

セットアップ、シスプレックス全体で同一 291
説明、Developer for System z 205
前提条件
 ホスト 357
 SMP/E 359
前提条件、Developer for System z 357
前提条件の製品 5
相互前提条件
 デバッグ・ツール 363
 ホスト 360
 Ant 368
 Apache Ant 368
 CICS Transaction Server 364
 COBOL コンパイラ 363
 DB2 365
 Fault Analyzer 367
 File Manager 367
 IMS 365
 Java 2 Technology Edition 359
 PL/I コンパイラ 363
 Ported Tools 368
 Rational Team Concert for System z 366
 REXX 367
 SDK for z/OS 359
 z/OS 360
相互前提条件、Developer for System z 357
双方向言語サポート 98
ソフトウェアの構成、必要な製品の 10
ソフトウェア・レベル、異なる構成ファイルで同一 292

[タ行]

代替 RSE 接続方式 110
代替セットアップ・オプション、APPC および VTAM 354
代替の CARMA サーバーの始動 65
代替パートナー LU 名、TSO コマンド・サービス 354
タスク、オプションのカスタマイズ 95
タスク所有者 208
ダンプ、Java 153
ダンプ、MVS 152
ダンプ・ファイル 152
ダンプ・ロケーション、z/OS UNIX 154
チェックリスト、クライアント 14
チェックリスト、要件 17
チェックリスト、APPC トランザクション 112
チェックリスト、SCLM 管理者 94
チューニングに関する考慮事項 227
調整、CRASERV.properties の 65
調整、ISPF.conf の 65

通信、外部 173
通信、内部 174
通信、SSL 暗号化 181, 273
通信暗号化、SSL を使用した 172
データ・セットのオープンの失敗、MVS 166
データ・セット・プロファイルの定義 192
テーブル、変換 332
テーブル、ローカル・ホスト 332
デーモン、ロック 26
デーモン、RSE 25
定義、セキュリティ 29, 189
定義、前提条件の LINKLIST および LPA 22
定義、リゾルバーで使用可能な 335
定義、RSE の MVS プログラム制御ライブラリー 199
定義、RSE の PassTicket サポート 200
定義、RSE の z/OS UNIX プログラム制御ファイル 201
ディスク・スペース、Java 仮想マシン (JVM) 266
ディレクトリー構造、z/OS UNIX グラフィカル表現 214
ディレクトリー・クリーンアップ、WORKAREA skulker 115
テスト、SSL ホスト構成接続の 321
テスト・ロギング、fekfivpi IVP 152
デバッグ・ツール
 相互前提条件 363
デプロイメント前の考慮事項 13
同一セットアップ、シスプレックス全体 291
同一のソフトウェア・レベル、異なる構成ファイル 292
統合、File Manager 106
特定のポート、外部通信の限定 171
トラブルシューティング、構成問題の 145
トランザクション名、TSO コマンド・サービス用の FEKFRSRV に対する代替 354
トランザクション・セキュリティ 272
トランザクション・ダンプ・パターン変数 153
トランザクション・チェックリスト、APPC 112
トレース 154
トレース、エラー・フィードバック 156
トレース、ロック・デーモン 156
トレース、CARMA 156
トレース、JES ジョブ・モニターの 154
トレース、RSE 102, 155
トレース構成、rsecomm.properties 102

[ナ行]

内部通信 174
認証、セキュリティ・ソフトウェアによる 184
認証、JES ジョブ・モニター 171
認証、RSE デーモンによる 185
認証、SSL および X.509 のセットアップ 315
認証局の妥当性検査
 gskkyman 183
 SAF 鍵リング 183
 TRUST、HIGHTRUST 183
認証方式 170
ネーム変換、SCLM 90
ネットワークのモニター 255

[ハ行]

バージョン 7.0 とバージョン 7.1 の間の
 変更点 311
パイプライン・セキュリティ 271
パイプライン・メッセージ・ハンドラー
 84
場所、秘密鍵と証明書を保管する 316
パスワードおよびユーザー ID 170
バックアップ、構成したファイル
 バージョン 7.0 297
 バージョン 7.1 297
 バージョン 7.5 297
バッチ実行依頼、CA Endeavor® SCM
 RAM の始動に使用 70
バッチ実行依頼、CARMA サーバーの始
 動に使用 60
バッチ実行依頼を使用した CARMA サー
 バーの始動 60
パフォーマンスに関する考慮事項 261
パフォーマンスの向上、セキュリティ検
 査の 263
ハンドラー、パイプライン・メッセージ
 84
ヒープ・サイズの限度、Java 239
非英数字、構文図 144
非システム管理者の更新特権 215
非主接続領域、CICS 82
必要条件、Developer for System z 357
必要条件に関する問題、既知 166
必要な製品およびソフトウェアの構成 10
必要な製品およびソフトウェアの必須の構
 成 10
必要なリソース 6
秘密鍵と証明書を保管する場所の決定
 316
ファイル・システム、zFS 261
ファイル・システム属性、SETUID 157
ファイル・システム・スペースの使用量、
 z/OS UNIX 244
フィードバック・トレース、エラー 156
複数の APPC トランザクション 288
複数の Developer for System z セットア
 ップでの複数の APPC トランザクシ
 ョンの使用 288
複数の Developer for System z セットア
 ップでの複数の ISPF.conf ファイルの使
 用 286
複数の ISPF.conf ファイル 286
複数の LU 354
複数のインスタンスの実行 291
複数の割り振り exec の使用、
 TSO/ISPF 285
フラグメント、構文 144
ブランク・スペース、構文図 144
プログラム制御許可 158
プロジェクト、ホスト・ベース 105
プロセスの数 232
プロパティ・グループ、ホスト・ベース
 104
プロファイルの定義、データ・セット
 192
分類規則、WLM 218
変換テーブル 332
変更、ワークロード・マネージャーの 95
変更、DB2 の 97
変更、PROCLIB の 96
変更点、バージョン 7.0 とバージョン 7.1
 の間の 311
編集不可能文字、処理するためのカスタマ
 イズ 108
編集不可能文字、uchars.settings 108
ポート、予約済み TCP/IP 161
ポート、CARMA と TCP/IP 174
ポート、REXEC 110
ポート、TCP/IP 173
ポート可用性 121
ポート定義、PROFILE.TCPIP 342
方式、認証 170
ホスト
 前提条件 357
ホストの相互前提条件 360
ホストの前提条件 357
ホスト名、Developer for System z への適
 用 333
ホスト名依存関係 329
ホスト・コンポーネントのインストールと
 構成 3
ホスト・テーブル、ローカル 332
ホスト・ベース・プロジェクト 105
ホスト・ベース・プロパティ・グループ
 104

[マ行]

マイグレーション 297
マイグレーション、バージョン 7.5 から
 7.6 へ 301
マイグレーションに関する考慮事項 3,
 297
マイグレーションに関する注、管理ユーテ
 ィリティー 277
前に構成したファイル、バックアップ
 バージョン 7.0 297
 バージョン 7.1 297
 バージョン 7.5 297
マニフェスト・リポジトリ 86
メッセージ、管理ユーティリティー 278
メッセージ、コンソール 140
メッセージ・ハンドラー、パイプライン
 84
目標、WLM での設定 219
目標の設定、WLM 219
文字、構文図の非英数字 144
モニター、ネットワーク 255
モニター、RSE 252
モニター、z/OS UNIX 253
モニター、z/OS UNIX ファイル・システ
 ム 256

[ヤ行]

ユーザー ID およびパスワード 170
ユーザー ID およびワнтаイム・パスワ
 ード 170
ユーザー ID に関する考慮事項 10
ユーザー・ロギング、RSE 148
要件、始動 JCL の 162
要件、チェックリスト 17
読み方、構文図の 143
予約済み TCP/IP ポート 161

[ラ行]

ライブラリー、言語環境プログラム・ラン
 タイム 262
ライブラリーの定義、RSE の MVS 199
ライブラリーへのアクセスの改善、システ
 ム 262
ランタイム・ライブラリー、言語環境プロ
 グラム 262
リソース、必要な 6
リソース使用量、概要 228
リソース使用量、チューニング 227
リソース定義、さまざまな 250
リソース・インストール・ロギング、
 CICS 271
リソース・セキュリティ 273

リゾルバーで使用可能なローカル定義 335
リゾルバーについて 330
リポジトリ、マニフェスト 86
リポジトリ、CRD 80
リポジトリ・セキュリティ、CRD 271
リモート実行、使用 109
リモート・システム・エクスプローラー 11
リモート・ビルド・プロシージャ、ELAXF* 27
リモート・ホスト・ベース・アクション、z/OS UNIX サブプロジェクト 109
ローカル・ホスト・テーブル 332
ロギング、スレッド・プール 148
ロギング、ロック・デーモン 147
ロギング、APPC トランザクション 151
ロギング、CARMA 150
ロギング、Fault Analyzer Integration 150
ロギング、fekfivpi IVP テスト 152
ロギング、File Manager Integration 150
ロギング、JES ジョブ・モニター 147
ロギング、RSE デーモン 148
ロギング、RSE ユーザー 148
ロギング、SCLM Developer Toolkit 150
ロギング構成ファイル、rsecomm.properties 102
ログとセットアップの分析に使用、FEKLOGS 145
ログ・ファイル
audit.log 146
fa.log 146
fekfivpi.log 146
fekfivps.log 146
ffsget.log 146
ffsput.log 146
ffs.log 146
lock.log 146
rmt_class_loader.cache.jar 146
rsecomm.log 146
rsedaemon.log 146
rseserver.log 146
serverlogs.count 146
stderr.log 146
stdout.log 146
.dstoreMemLogging 146
.dstoreTrace 146
ロックの解放
RSE、modify cancel コマンド 213
ロック・デーモン 26, 212
コンソール・メッセージ 140
ロック・デーモン (LOCKD) 205
ロック・デーモン、LOCKD 117
ロック・デーモン、Modify コマンド 139
ロック・デーモン、Start コマンド 134

ロック・デーモン接続 124
ロック・デーモンのトレース 156
ロック・デーモンのフロー
グラフィカル表現 212
ロック・デーモン・ロギング 147
ロング/ショート・ネーム変換、SCLM 90
ロング/ショート・ネーム変換用のrsed.envvars の更新 92

[ワ行]

ワークロード管理 263
ワークロード分類、WLM 217
ワークロード・マネージャー 217
ワークロード・マネージャーの変更 95
割り振り exec (APPC トランザクション JCL)、使用 288
割り振り exec の使用 285
ワнтаイム・パスワードおよびユーザー ID 170

A

ADM のカスタマイズ 79
admin、SCLMDT 93
ADNCSDAR、RESTful 非主と Web サービス非主 82
ADNCSDRS、RESTful 主 82
ADNCSDTX、RESTful での ID の変更 83
ADNCSDWS、Web サービス主 85
ADNJSPAU、管理ユーティリティ 273
ADNJSPAU、CICS 管理ユーティリティ 81
ADNMSGHC、パイプライン・メッセージ・ハンドラー 84
ADNMSGHS、パイプライン・メッセージ・ハンドラー 84
ADNTXNC、RESTful での ID の変更 83
ADNVCRD、CRD リポジトリ 80
ADNVMFST、マニフェスト・リポジトリ 86
Ant
相互前提条件 368
Ant のインストールおよびカスタマイズ 92
Apache Ant
相互前提条件 368
APF 許可
FEK.SFEKAUTH 193
APF、許可 159
APPC アクセス方式の使用 287
APPC 代替セットアップ・オプション 354

APPC トランザクション
APPC のトラブルシューティング 164
APPC トランザクション JCL での既存の ISPF プロファイルの使用 287
APPC トランザクション JCL、基本カスタマイズ 287
APPC トランザクション JCL、割り振り exec の使用 288
APPC トランザクション、実装 113
APPC トランザクション、準備 111
APPC トランザクション、複数の Developer for System z でのセットアップ 288
APPC トランザクションおよび TSO コマンド・サービス 164
APPC トランザクション・チェックリスト 112
APPC トランザクション・ロギング 151
APPC の使用に関する考慮事項 114
APPC のセットアップ 349
APPC の変更のアクティブ化 353
APPCPMxx 351
Application Deployment Manager (ADM) 205
Application Deployment Manager セキュリティー 271
Application Deployment Manager のカスタマイズ 79
Application Deployment Manager、カスタマイズ 269
Application Deployment Manager、CICS リソース定義エディター 269
Application Deployment Manager、CICS リソース定義サーバー 269
ASCHPMxx 352
MAX 252
ASSIZEMAX 196
audit.log 147

B

BPXBATCH、INETD の始動 344
BPXPRMxx 258
INADDRANYCOUNT 250
MAXASSIZE 162, 196, 249
MAXFILEPROC 249
MAXMMAPAREA 249
MAXPROCSYS 166, 248
MAXPROCUSER 166, 248
MAXSOCKETS 250
MAXTHREADS 248
MAXTHREADTASKS 248
MAXUIDS 166, 249
BPXPRMxx での限度の設定
MAXASSIZE 19
MAXPROCUSER 19

BPXPRMxx での限度の設定 (続き)

MAXTHREADS 19

MAXTHREADTASKS 19

C

CA Endeavor® SCM RAM 68

CA Endeavor® SCM RAM の始動、バッチ
実行依頼 70

CA Endeavor® SCM RAM の始動、
CRASTART 72

CA Endeavor® SCM RAM、カスタマイズ
75

CARMA サーバーの始動、代替の 56,
62, 65

CARMA と TCP/IP ポート 174

CARMA トレース 156

CARMA のアクティブ化 55

CARMA の始動、バッチ実行依頼 60

CARMA への RSE インターフェース 58

CARMA ログイン

rsecomm.log 150

CARMA, FMID HCMA710 311

CEE.SCEELPA

SYS1.PARMLIB(LPALSTxx) 262

CICS Transaction Server

相互前提条件 364

CICS 管理者向け管理ユーティリティ
提供されている機能 273

CICS 管理ユーティリティ 81

CICS 主接続領域 82, 85

CICS 双方向言語サポート 98

CICS トランザクション 187

CICS 非主接続領域 82, 85

CICS リソース定義 (CRD) エディター、
Application Deployment Manager 269

CICS リソース定義 (CRD) サーバー、
Application Deployment Manager 269

CICS リソース定義、開発者 269

CICS リソース定義、管理者 269

CICS リソース・インストール・ロギング
271

CICSplex SM ビジネス・アプリケーション・
サービス (BAS) 270

CICSTS セキュリティ 187

CICSTS に関する考慮事項 269

CLASSPATH 292

COBOL

リモート検査 156

COBOL コンパイラー

相互前提条件 363

COMMNDxx、開始タスクの追加 20

CRAISPRX、ISPF 構成ファイル 51

CRAISPRX、ISPF 代替 CARMA 始動
65

CRANDVRA、カスタマイズ 74

CRASERV.properties の調整 65

CRASRV.properties

clist.dsname 58

crastart.configuration.file 58

crastart.steplib 58

crastart.stub 58

crastart.syslog 58

crastart.tasklib 58

crastart.timeout 58

port.range 58

port.start 58

startup.script.name 58

CRASRV.properties、調整 60, 62, 70, 72

CRASTART、代替の CARMA サーバーの
始動に使用 62

CRASTART、CA Endeavor® SCM RAM の
始動に使用 72

crastart.conf、調整 62

crastart.conf、CARMA crastart 始動 62

crastart.endeavor.conf、調整 73

CRASUBCA、調整 71

CRASUBMT、CARMA バッチ始動 61

CRAXJCL、IRXJCL と CRAXJCL 77

CRA#ASLM、サンプルの SCLM

RAM 68

CRA#CIRX、IRXJCL と CRAXJCL 77

CRA#CRAM、サンプルのスケルトン

RAM 68

CRA#UADD、CARMA コンポーネント

56, 69

CRA#UQRY、CARMA コンポーネント

56, 69

CRA#VCAD、Endeavor SCM RAM 69

CRA#VCAS、Endeavor SCM RAM 69

CRA#VPDS、サンプルの PDS RAM 67

CRA#VSLM、サンプルの SCLM

RAM 68

CRA\$VDEF、CARMA コンポーネント

56

CRA\$VMSG、CARMA コンポーネント

56, 69

CRA\$VSTR、CARMA コンポーネント

56

CRD サーバーが使用、RESTful インター
フェース 82

CRD サーバー・トランザクション ID、
カスタマイズ 83

CRD リポジトリ 80, 187

CRD リポジトリ・セキュリティ

271

cron、WORKAREA ディレクトリー・ク
リーンアップ 115

D

DB2

相互前提条件 365

DB2 ストアード・プロシージャー 95

DB2 の変更 97

Debug Tool、IBM、相互前提条件 23

Developer for System z 開始タスク、定義
196

Developer for System z について 205

Developer for System z、コンポーネント
の概要

グラフィカル表現 205

E

ELAXF* 高位修飾子チェックリスト 29

ELAXF* プロシージャー、サンプル

ELAXFADT 27

ELAXFASM 27

ELAXFBMS 27

ELAXFCOC 27

ELAXFCOP 27

ELAXFCOT 27

ELAXFCPI 27

ELAXFCPC 27

ELAXFCPP 27

ELAXFDCL 27

ELAXFGO 27

ELAXFLNK 27

ELAXFMFS 27

ELAXFPL1 27

ELAXFPLP 27

ELAXFPLT 27

ELAXFPP1 27

ELAXFTSO 27

ELAXFUOP 27

ELAXF* リモート・ビルド・プロシージ
ャー 27

ELAXMJCL、DB2 の変更 97

ELAXMSAM 96

Emulator、Host Connect 167

EST サポート 97

ETC.SERVICES

aliases 341

port_number 341

protocol 341

service_name 341

F

Fault Analyzer

相互前提条件 367

Fault Analyzer Integration ログイン

fa.log 150

rsecomm.log 150

fa.log 146
 FEJJC�FNG 174, 258, 294
 CONSOLE_NAME 179
 MAX_THREADS 251
 FEJJC�FNG, JES ジョブ・モニター 188, 189
 FEJJC�FNG, JES ジョブ・モニター構成ファイル 30
 FEJJCL, PROCLIB の変更、JES ジョブ・モニター 24
 FEJTSO 30
 FEKAPPCL, APPC 実装 113
 FEKAPPX, APPC 実装 113
 FEKAPPC, APPC 実装 113
 FEKAPPL 171
 fekfivpi IVP テスト・ロギング
 fekfivpi.log 152
 fekfivpi.log 146
 fekfivpi.log, IVP テスト・ロギング 152
 fekfivps.log, IVP テスト・ロギング 152
 FEKFRSRV 114
 FEKLOCKD 26
 FEKLOGS, ログとセットアップの分析に使用 145
 FEKRCF, セキュリティ定義 29, 189
 fekrivp 159
 FEKRSED 25
 FEKSETUP 18, 82, 88, 113
 ffsget.log 146
 ffsput.log 146
 ffs.log 146
 File Manager
 相互前提条件 367
 File Manager Integration 106
 File Manager Integration ロギング
 rsecomm.log 150
 FMID HCMA710 311
 FMID HHOP710 311
 FMID HHOP750 307
 FMID HHOP760 301
 FMIEXT.properties
 enabled 107
 fmlistenport 107

G

gskkyman による鍵データベースの作成 325

H

HCMA710 311
 HHOP710 311
 HHOP750 307
 HHOP760 301

Host Connect Emulator 167

I

IBM Debug Tool、相互前提条件 23
 ID に関する考慮事項、ユーザー 10
 IEASYSxx 258
 MAXUSER 166, 251
 IMS
 相互前提条件 365
 INETD 一時ファイル /etc/inetd.pid 343
 INETD 環境設定 347
 INETD セキュリティ 345
 INETD のセットアップ 339
 INETD, Developer for System z の要件 347
 inetd.conf 339
 protocol 339
 server_program 339
 server_program_arguments 339
 service_name 339
 socket_type 339
 userid 339
 wait_flag 339
 IRXJCL, IRXJCL と CRAXJCL 77
 ISPF TSO/ISPF クライアント・ゲートウェイ
 ISP.SISPLOAD 199
 ISPF コマンド 52, 65
 ISPF の TSO/ISPF クライアント・ゲートウェイ接続の検査 124
 ISPF プロファイル (APPC トランザクション JCL) 287
 ISPF プロファイルの使用、既存の 284
 ISPF、複数の割り振り exec の使用 285
 ISPF.conf 51
 allocjob 52
 ISPF_timeout 52
 isplib 52
 ispmlib 52
 isplib 52
 ispslib 52
 isptlib 52
 sysproc 52
 ISPF.conf の更新 SCLMDT 用の 88
 ISPF.conf の調整 65
 ISPF.conf ファイル、複数のセットアップでの使用 286
 ISPF.conf、基本カスタマイズ 284
 ISP.SISPLOAD
 ISPF TSO/ISPF クライアント・ゲートウェイ 199
 IVP
 fekfivpa 126
 fekfivpd 123
 fekfivpi 124

IVP (続き)

 fekfivpj 123
 fekfivpl 124
 fekfivpr 128
 fekfivps 126
 fekfivpz 129
 IVP テスト・ロギング
 fekfivpi.log 152
 fekfivps.log 152
 IVP の初期化
 ivpinit 120
 ivpinit シェル・スクリプト、RSE 環境変数の設定 120
 IVP、基本およびオプションのサービス
 fekfivpa 120
 fekfivpd 120
 fekfivpi 120
 fekfivpj 120
 fekfivpl 120
 fekfivpr 120
 fekfivps 120
 fekfivpt 120
 fekfivpz 120
 IVP スクリプト 120
 IVTPRMxx
 ECSA MAX 251
 FIXED MAX 251

J

J2EE 87, 92
 Java 87, 92
 Java 2 Technology Edition
 相互前提条件 359
 Java Xquickstart オプション 264
 Java アプリケーションとしての RSE
 グラフィカル表現 207
 Java 仮想マシン (JVM) 間のクラス共用 264
 Java 始動パラメーター、
 _RSE_CMDSEV_OPTS での定義 51
 Java 始動パラメーター、
 _RSE_JAVAOPTS での定義 46
 Java ダンプ 153
 Java ヒープ・サイズ、固定 264
 Java ヒープ・サイズの限度 239
 JAVA_DUMP_TDUMP_PATTERN 153
 JCL の要件、始動 162
 JES JMON
 GEN_CONSOLE_NAME 180
 JES JMON, FEJJC�FNG
 APPLID 31
 AUTHMETHOD 31
 CODEPAGE 31
 CONCHAR 31
 CONSOLE_NAME 31

JES JMON、FEJCNFG (続き)

GEN_CONSOLE_NAME 31
HOST_CODEPAGE 31
LIMIT_COMMANDS 31
LIMIT_VIEW 31
LISTEN_QUEUE_LENGTH 31
MAX_DATASETS 31
MAX_THREADS 31
SERV_PORT 31
SUBMITMETHOD 31
TIMEOUT 31
TIMEOUT_INTERVAL 31
TSO_TEMPLATE 31
TZ 31
_BPXK_SETIBMOPT
_TRANSPORT 31

JES コマンド・セキュリティの定義 197

JES ジョブ・モニター (JMON) 205

JES ジョブ・モニター、FEJCNFG 188

JES ジョブ・モニター、JMON 117

JES ジョブ・モニター、Modify コマンド 135

JES ジョブ・モニター、Start コマンド 133

JES ジョブ・モニター開始タスク、JMON 24

JES ジョブ・モニター構成
GEN_CONSOLE_NAME 180

JES ジョブ・モニター構成ファイル、FEJCNFG、 30

JES ジョブ・モニター接続 123

JES ジョブ・モニターのトレース 154

JES ジョブ・モニターの認証 171

JES ジョブ・モニター・サーバー 24

JES ジョブ・モニター・ロギング 147

JES セキュリティー 177

JMON 24, 197, 294

fekfivpj 123

JMON、JES ジョブ・モニター 117

JVM 間のクラス共用 264

K

keytool による鍵ストアの作成 327

L

LIMIT_COMMANDS 178

LIMIT_VIEW 180

LINKLIST 定義、前提条件の 22

LINKLIST、他の製品用の定義 23

LOCKD 11

LOCKD、ロック・デーモン 117

lock.log 146

LPA 定義、前提条件の 22

LPALSTxx 262

LPALSTxx、LPA 定義 20

LSTRANS.FILE、作成 90

LU セキュリティー、VTAM 355

M

Modify (F) コマンド 135

MVS ダンプ 152

MVS データ・セットのオープンの失敗 166

MVS プログラム制御ライブラリーの定義、RSE の 199

N

netstat 161

netstat、TCP/IP のセットアップ 122

O

OMVS セグメントの定義 192

P

PARM 変数、JCL での制限 26

PARMLIB、変更 19

PassTicket サポートの定義、RSE の 200

PassTicket の使用 175

PDS Repository Access Manager のアクティブ化 67

PL/I コンパイラー

相互前提条件 363

POE 検査 172, 186

Port Of Entry 検査 172, 186

Port Of Entry 検査の定義、RSE 用 186

Ported Tools

相互前提条件 368

PORTRANGE 45, 161

PORTRANGE の定義、RSE サーバーに使用可能な 45

PROCLIB の変更 24, 96

PROFILE.TCPIP ポート定義 342

PROGxx での LINKLIST 定義 21

PROGxx、APF 許可 21

projectcfg.properties 105

PROJECT-HOME 106

WSED-VERSION 106

propertiescfg.properties 104

DEFAULT-VALUES 105

ENABLED 105

PROPERTY-GROUP 105

RDZ-VERSION 105

R

RACF

許可 194

RACF による鍵リングの作成 317

Rational Team Concert for System z

相互前提条件 366

Repository Access Manager (RAM) のアクティブ化、サンプルの 67

Repository Access Manager のアクティブ化、スケルトン 68

Repository Access Manager のアクティブ化、PDS 67

Repository Access Manager のアクティブ化、SCLM 68

Repository Access Manager の略。 68

Repository Access Manager、CA Endeavor® SCM のアクティブ化 68

RESTful インターフェース 270

ADMI 83

ADMR 83

ADMS 83

RESTful インターフェースと Web サービス・インターフェース 81, 270

RESTful インターフェースを使用する
CRD サーバー 82

REXEC 接続の検査 128

REXEC デーモン、INETD によって始動されるときユーザー ID 権限 347

REXEC のセットアップ 110

REXEC、使用 109

REXEC/SSH シェル・スクリプト 129

REXX

相互前提条件 367

rmt_class_loader_cache.jar 146

RSE 11

RSE SSL 暗号化、ssl.properties、
サーバーのプロパティ 100
デーモンのプロパティ 100

RSE インターフェース、CARMA への 58

RSE 開始タスク、RSED 25

RSE 構成ファイル、rsed.envvars、 35

RSE サーバー 173

RSE サーバーでのスレッド・セキュリティ

PassTicket 175

RSE サーバーの定義、セキュアな z/OS UNIX として 199

RSE スレッド・プールのログ・ファイル
audit.log 148

rsedaemon.log 148

rseserver.log 148

serverlogs.count 148

stderr.*.log 148

stdout.*.log 148

RSE スレッド・プール・サーバー
 コンソール・メッセージ 140

RSE 接続方式、代替 110

RSE セットアップのクローン作成、既存
 の 319

RSE デーモン 11, 25, 173
 コンソール・メッセージ 140

RSE デーモン (RSED) 205

RSE デーモン、Modify コマンド 136

RSE デーモン、RSED 117

RSE デーモン、Start コマンド 134

RSE デーモン接続 123

RSE デーモンと監査ロギング 176

RSE デーモンによる認証 185

RSE デーモンのログ・ファイル
 audit.log 148
 rsedaemon.log 148
 rseserver.log 148
 serverlogs.count 148
 stderr.*.log 148
 stdout.*.log 148

RSE デーモン・ロギング 148

RSE トレース 155

RSE トレース構成、
 rsecomm.properties、 102

RSE のモニター 252

RSE ユーザー・ロギング
 ffsget.log 148
 ffspu.log 148
 ffs.log 148
 lock.log 148
 rmt_class_loader.cache.jar 148
 rsecomm.log 148
 stderr.log 148
 stdout.log 148
 .dstoreMemLogging 148
 .dstoreTrace 148

rsecomm.log 146
 File Manager Integration ロギング 150
 SCLM Developer Toolkit のロギング
 150

rsecomm.properties 155
 debug_level 103
 log_location 103
 server.version 103

rsecomm.properties、 102

RSED 25

rsedaemon.log 146, 147

RSED、RSE デーモン 117

rsed.envvars 136, 246, 292
 ANT_HOME 41
 CGI_DTWORK 43
 CLASSPATH 43
 DAPPLID 50
 Daudit.cycle 48
 Daudit.retention.period 48

rsed.envvars (続き)
 Ddaemon.log 46
 Ddeny.nonzero.port 49
 DDENY_PASSWORD 50
 DDSTORE_IDLE_SHUTDOWN
 _TIMEOUT 50
 DDSTORE_LOG_DIRECTORY 46
 DDSTORE_MEMLOGGING_ON 50
 DDSTORE_TRACING_ON 50
 Denable.audit.log 48
 Denable.automount 48
 Denable.certificate.mapping 48
 Denable.port.of.entry 48
 Denable.standard.log 48
 DHIDE_ZOS_UNIX 50
 Dipv6 48
 Dkeep.last.log 48
 Dmaximum.clients 46, 247
 Dmaximum.threadpool.process 48, 247
 Dmaximum.threads 46, 247
 Dminimum.threadpool.process 46, 247
 Dprocess.cleanup.interval 49
 Dsingle.logon 49
 DSTORE_LOG_DIRECTORY 150, 155
 DTSO_SERVER 50
 Duser.log 46
 GSK_CRL_SECURITY_LEVEL 41
 GSK_LDAP_PASSWORD 41
 GSK_LDAP_PORT 41
 GSK_LDAP_SERVER 41
 GSK_LDAP_USER 41
 JAVA_HOME 39
 JAVA_PROPAGATE 43
 LANG 39
 LIBPATH 43
 PATH 39, 43
 RSE_HOME 39
 RSE_JAVAOPTS 40
 RSE_LIB 43
 RSE_SAF_CLASS 40
 SCLMDT_BASE_HOME 43
 SCLMDT_CONF_HOME 41
 SCLMDT_WORK_HOME 43
 STEPLIB 39, 40, 41, 182
 TZ 39
 Xms 46, 247
 Xmx 46, 247
 &ISPROF=&SYSUID..ISPPROF= 51
 _BPXK_SETIBMOPT
 _TRANSPORT 41
 _BPX_SHAREAS 43
 _BPX_SPAWN_SCRIPT 43
 _CEE_DMPTARG 39
 _CEE_RUNOPTS 43
 _CMDSERV_BASE_HOME 40
 _CMDSERV_CONF_HOME 40, 286

rsed.envvars (続き)
 _CMDSERV_WORK_HOME 40
 _FEKFSCMD_PARTNER_LU_ 41
 _FEKFSCMD_TP_NAME_ 41
 _RSE_CMDSERV_OPTS 43
 _RSE_DAEMON_CLASS 43
 _RSE_HOST_CODEPAGE 39
 _RSE_JAVAOPTS 43, 153, 283
 _RSE_LOCKD_CLASS 43
 _RSE_LOCKD_PORT 39
 _RSE_POOL_SERVER_CLASS 43
 _RSE_PORTRANGE 41, 171
 _RSE_SERVER_CLASS 43
 _RSE_SERVER_TIMEOUT 43
 _SCLMDT_TRANTABLE 41

rsed.envvars の更新、共存を可能にするた
 めの 319

rsed.envvars の更新、SCLMDT 用の 89

rsed.envvars、ロング/ショート・ネーム変
 換用に更新 92

rsed.envvars、RSE 構成ファイル 35

rseserver.log 146, 147

RSE、アプリケーション保護の定義 200

RSE、セキュアな z/OS UNIX サーバーと
 して定義 199

RSE、MVS プログラム制御ライブラリー
 の定義 199

RSE、PassTicket サポートの定義 200

RSE、Port Of Entry 検査の定義 186

RSE、PORTRANGE の定義 45

RSE、rsed.envvars
 _RSE_JAVAOPTS 188

RSE、ssl.properties 189

RSE、z/OS UNIX プログラム制御ファイ
 ルの定義 201

S

SCLM 92

SCLM Developer Toolkit 199

SCLM Developer Toolkit (SCLMDT) 205

SCLM Developer Toolkit サービス 22

SCLM Developer Toolkit 接続の検査
 SCLMDT 検査 126

SCLM Developer Toolkit のカスタマイズ
 87

SCLM Developer Toolkit のロギング
 rsecomm.log 150

SCLM Repository Access Manager のアク
 ティブ化 68

SCLM 管理者チェックリスト 94

SCLM セキュリティ 187

SCLMDT admin 93

SCLMDT 接続の検査 126

SCLMDT 用の ISPF.conf の更新 88

SCLMDT 用の rsed.envvars の更新 89

SCLMDT 用の SCLM の更新 93
SCLM、ロング/ショート・ネーム変換 90
SDSF 88
Secure Sockets Layer のセットアップ 315
Secure Sockets Layer ホスト構成接続のテスト 321
Secure Sockets Layer を使用した通信暗号化 172
serverlogs.count 146
SETUID ファイル・システム属性 157
skulker、WORKAREA ディレクトリー・クリーンアップ 115
SMP/E
 前提条件 359
SMP/E インストール、スティッキー・ビット 160
Software Configuration Manager 68
SSH シェル・スクリプト 129
SSH デーモン、INETD によって始動されるときユーザー ID 権限 347
SSH のセットアップ 110
SSH、使用 109
SSL 暗号化通信 181, 187, 273
SSL のセットアップ 315
SSL ホスト構成接続のテスト 321
SSL を使用した通信暗号化 172
ssl.properties 100
 daemon_keydb_file 101
 daemon_keydb_password 101
 daemon_key_label 101
 enable_ssl 101
 server_keystore_file 101
 server_keystore_label 101
 server_keystore_password 101
 server_keystore_type 101
ssl.properties の更新による SSL のアクティブ化 320
ssl.properties、新しい RSE デーモンの作成による SSL のアクティブ化 320
Start (S) コマンド 133
stderr.log 146
stderr.*.log 146
stdout.log 146
stdout.*.log 146
STEPLIB の使用 23
STEPLIB の使用の回避 261
Stop (P) コマンド 140
SYS1.PARMLIB(APPCPMxx) 351
SYS1.PARMLIB(ASCHPMxx) 352
SYS1.PARMLIB(BPXPRMxx) 258
 MAXASSIZE 162, 196
 MAXPROCSYS 166
 MAXPROCUSER 166
 MAXUIDS 166

SYS1.PARMLIB(BPXPRMxx) で設定される制限 162
SYS1.PARMLIB(BPXPRMxx)、Java 仮想マシン (JVM) 266
SYS1.PARMLIB(IEASYSxx) 258
 MAXUSER 166
SYSEXEC 52
SYSPROC 52

T

TCP/IP のセットアップ 329
TCP/IP のセットアップ、netstat 122
TCP/IP ポート 173
TCP/IP ポート、グラフィカル表現 173
TCP/IP ポート、予約済み 161
TCP/IP リゾルバー、解決されないホスト・アドレス
 lock.log 336
TCP/IP、リゾルバーで使用可能なローカル定義 335
TCP/IP、Developer for System z への適用 333
TN3270 363
TSO アクセス方式 283
TSO 環境のカスタマイズ 283
TSO コマンド 65
TSO コマンド・サービス 164, 205, 283
TSO コマンド・サービス接続 (APPC) 126
TSO コマンド・サービス用の APPC トランザクション 111
TSO コマンド・サービス・トランザクションの定義 354
TSO コマンド・サービス・ロギング 151
TSO/ISPF クライアント・ゲートウェイ構成ファイル 51
TSO/ISPF クライアント・ゲートウェイ・アクセス方式の使用 284
TSO/ISPF、カスタマイズ -
 ISPF.conf 284
TSO/ISPF、既存の ISPF プロファイルの使用 284
TSO/ISPF、複数のセットアップでの使用 286
TSO/ISPF、複数の割り振り exec の使用 285
TSO/ISPF、割り振り exec の使用 285

U

uchars.settings、編集不可能文字 108
UNIX 環境で使用される検索順序 331
UNIX サーバーとしての RSE の定義 199

UNIX ダンプ・ロケーション 154
UNIX プログラム制御ファイルの定義、RSE の 201

V

VSAM 349
VTAM 350
VTAM 代替セットアップ・オプション 354

W

Web Owning Region 270
Web サービス・インターフェース 270
 ADMI 84
 ADMR 84
 ADMS 84
Web サービス・インターフェース、CRD サーバーが使用 83
Web サービス・インターフェースと RESTful インターフェース 81
WLM に関する考慮事項 xvii, 217
WLM 分類規則 218
WORKAREA からの古いファイルの除去 94
WORKAREA ディレクトリー・クリーンアップ
 skulker 115

X

Xquickstart、Java オプション 264
X.509 クライアント認証サポートの追加 324
X.509 証明書 170
X.509 証明書を使用したクライアント認証 182
x.509 認証、セットアップ 315

Z

zFS ファイル・システムの使用 261
z/OS
 相互前提条件 360
z/OS UNIX 環境で使用される検索順序 331
z/OS UNIX 許可ビット 157
z/OS UNIX サーバーとしての RSE の定義 199
z/OS UNIX サブプロジェクト用のリモート・ホスト・ベース・アクション 109
z/OS UNIX ダンプ・ロケーション 154
z/OS UNIX ディレクトリー構造
 グラフィカル表現 214

z/OS UNIX の限度の設定、BPXPRMxx
での 19
z/OS UNIX のモニター 253
z/OS UNIX ファイル・システムのモニター 256
z/OS UNIX ファイル・システム・スペースの使用量 244
z/OS UNIX プログラム制御ファイルの定義、RSE の 201
z/OS の相互前提条件
言語環境プログラム 362
高水準アセンブラー 362
C/C++ 362
SCLM 362
z/OS プロジェクト・パースペクティブ
105
z/OS ホストの前提条件 357
z/OSホスト
前提条件 357

[特殊文字]

.dstoreMemLogging 146
.dstoreTrace 146
/etc/inittab、z/OS UNIX の初期化、
INETD 344
/etc/rc、z/OS UNIX INETD の始動 344
_RSE_CMDSERV_OPTS での追加 Java 始
動パラメーターの定義 51
_RSE_JAVAOPTS での追加 Java 始動パ
ラメーターの定義 46
_RSE_PORTRANGE 171



プログラム番号: 5724-T07

Printed in Japan

SC88-5663-02



日本アイ・ビー・エム株式会社

〒103-8510 東京都中央区日本橋箱崎町19-21