

IBM Communications Server for AIX



LUA プログラマーズ・ガイド

V6.3

IBM Communications Server for AIX



LUA プログラマーズ・ガイド

V6.3

ご注意

本書および本書で紹介する製品をご使用になる前に、183 ページの『付録 C. 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 Communications Server for AIX バージョン 6.3 プログラム番号 5765-E51 に適用されます。また、新しい版あるいはテクニカル・ニュースレターで特に断りのない限り、それ以降のリリースおよび変更にも適用されません。

資料のご注文方法については、<http://www.ibm.com/jp/manuals> の「ご注文について」をご覧ください。(URL は、変更になる場合があります)

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。

本マニュアルに関するご意見やご感想は、次の URL からお送りください。今後の参考にさせていただきます。

<http://www.ibm.com/jp/manuals/main/mail.html>

なお、日本 IBM 発行のマニュアルはインターネット経由でもご購入いただけます。詳しくは

<http://www.ibm.com/jp/manuals/> の「ご注文について」をご覧ください。

(URL は、変更になる場合があります)

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： SC31-8592-02
IBM Communications Server for AIX
LUA Programmer's Guide
V6.3

発 行： 日本アイ・ピー・エム株式会社

担 当： ナショナル・ランゲージ・サポート

第1刷 2005.10

この文書では、平成明朝体™W3、平成明朝体™W7、平成明朝体™W9、平成角ゴシック体™W3、平成角ゴシック体™W5、および平成角ゴシック体™W7を使用しています。この(書体*)は、(財)日本規格協会と使用契約を締結し使用しているものです。フォントとして無断複製することは禁止されています。

注* 平成明朝体™W3、平成明朝体™W7、平成明朝体™W9、平成角ゴシック体™W3、平成角ゴシック体™W5、平成角ゴシック体™W7

© Copyright International Business Machines Corporation 2000, 2005. All rights reserved.

© Copyright IBM Japan 2005

目次

表	vii
図	ix
本書について	xi
本書の対象読者	xi
本書の使用法	xi
本書の構成	xi
表記上の規則	xii
シンボルの意味	xiii
新機能	xiii
詳細について	xiv
第 1 章 概念	1
LUA とは	1
使用するインターフェースの選択	2
LU とセッション	3
構成	5
LUA verb	5
RUI verb の概要	5
SLI verb の概要	6
verb の非同期完了	7
LUA 通信順序の例	8
LUA の互換性	11
第 2 章 LUA アプリケーションの設計と作成	13
AIX または Linux アプリケーションの LUA エントリー・ポイント	13
RUI 関数呼び出し	14
SLI 関数呼び出し	14
指定パラメーター	14
戻り値	14
使用法	15
verb の非同期完了時用のコールバック・ルーチン	16
Windows アプリケーションの LUA エントリー・ポイント	16
RUI	18
WinRUIStartup	19
WinRUI	21
WinRUIGetLastInitStatus	24
WinRUICleanup	27
GetLuaReturnCode	28
SLI	29
WinSLIStartup	30
WinSLI	32
WinSLICleanup	34
LUA verb の発行	35
SNA 情報	38
BIND 検査: RUI	38
BIND 検査: SLI	39
否定応答と SNA センス・コード	39
ペーシング	41

セグメント化	41
非標準ホストの応答/要求ヘッダー (RH) ビットの変更	41
肯定応答	41
チェーン終了までのデータの除去	42
1 次 RUI の SNA 情報	42
RUI 1 次側アプリケーションの作業	42
ペーシング	43
セグメント化	43
制約事項	43
肯定応答	43
チェーン終了までのデータの除去	44
構成情報	44
データ・リンク制御 (DLC)、ポート、およびリンク・ステーション (LS)	45
LU	45
LU プール (オプション)	45
AIX または Linux に関する考慮事項	45
LUA ヘッダー・ファイル	45
マルチプロセスと複数セッション	46
LUA アプリケーションのコンパイルとリンク	46
Windows に関する考慮事項	47
複数セッションと複数タスク	47
LUA プログラムのコンパイルとリンク	47
アプリケーションの終了	48
移植可能なアプリケーションの作成	48
第 3 章 LUA VCB 構造体	51
LUA verb 制御ブロック (VCB) の形式	51
LUA_VERB_RECORD データ構造	52
共通データ構造	52
特定データ構造	58
第 4 章 RUI verb	63
RUI_BID	63
指定パラメーター	63
戻りパラメーター	64
他の verb との関連	69
使用法と制約事項	70
RUI_INIT	70
指定パラメーター	70
戻りパラメーター	73
他の verb との関連	77
使用法と制約事項	78
RUI_INIT_PRIMARY	78
指定パラメーター	78
戻りパラメーター	79
他の verb との関連	82
使用法と制約事項	82
RUI_PURGE	83
指定パラメーター	83
戻りパラメーター	84
他の verb との関連	87
RUI_READ	87
指定パラメーター	87
戻りパラメーター	89
他の verb との関連	95
使用法と制約事項	96

RUI_REINIT	97
指定パラメーター	97
戻りパラメーター	98
他の verb との関連	100
使用法と制約事項	101
RUI_TERM	101
指定パラメーター	101
戻りパラメーター	102
他の verb との関連	105
RUI_WRITE	106
指定パラメーター	106
戻りパラメーター	108
他の verb との関連	113
使用法と制約事項	113
第 5 章 SLI verb	115
SLI_BID	115
指定パラメーター	115
戻りパラメーター	116
他の verb との関連	122
使用法と制約事項	122
SLI_CLOSE	123
指定パラメーター	123
戻りパラメーター	124
他の verb との関連	128
使用法と制約事項	129
SLI_OPEN	130
指定パラメーター	130
SLI エントリー・ポイントからの戻り値	134
戻りパラメーター	135
他の verb との関連	139
使用法と制約事項	140
SLI_PURGE	140
指定パラメーター	140
戻りパラメーター	141
他の verb との関連	145
SLI_RECEIVE	145
指定パラメーター	145
戻りパラメーター	147
他の verb との関連	154
使用法と制約事項	154
SLI_SEND	155
指定パラメーター	155
戻りパラメーター	157
他の verb との関連	163
使用法と制約事項	164
SLI_BIND_ROUTINE	165
指定パラメーター	165
戻りパラメーター	165
他の verb との関連	166
使用法と制約事項	166
SLI_SDT_ROUTINE	166
指定パラメーター	166
戻りパラメーター	167
他の verb との関連	167
使用法と制約事項	167

SLI_STSN_ROUTINE	167
指定パラメーター	168
戻りパラメーター	168
他の verb との関連	169
使用法と制約事項	169
第 6 章 LUA アプリケーションのサンプル	171
処理の概要	171
アプリケーションのテスト	173
ホスト要件	173
サンプル・アプリケーションの構成	173
サンプル・アプリケーションのコンパイルとリンク	173
サンプル・アプリケーションの実行	174
付録 A. 戻りコードの値	177
1 次戻りコード	177
2 次戻りコード	177
付録 B. アクセシビリティ	181
支援機能の使用	181
ユーザー・インターフェースのキーボード・ナビゲーション	181
z/OS の情報	181
付録 C. 特記事項	183
商標	185
参考文献	187
CS/AIX バージョン 6.3 の資料	187
IBM Communications Server for AIX バージョン 4.2 関連資料	189
IBM Redbooks	189
ブロック・マルチプレクサーおよび S/390 ESCON チャンネル PCI アダプター 関連資料	190
AnyNet/2 ソケットおよび SNA 関連資料	190
AIX オペレーティング・システム関連資料	190
システム・ネットワーク体系 (SNA) 関連資料	190
ホスト構成関連資料	191
z/OS Communications Server 関連資料	191
マルチプロトコル・トランスポート・ネットワーキング関連資料	192
TCP/IP 関連資料	192
X.25 関連資料	192
APPC 関連資料	192
プログラミング関連資料	192
その他の IBM ネットワーキング関連資料	193
索引	195

表

1. 表記上の規則 xii
2. メッセージ・タイプに応じた SLL_SEND パラメーターの設定値. 164



1.	LUA 通信に使用される SNA コンポーネント	3
2.	1 次 RUI 通信に使用される SNA コンポーネント	4
3.	RUI 通信順序	10
4.	SLI 通信順序	11
5.	サンプルの LUA アプリケーションのプログラムの流れ	172

本書について

本書は、従来型の論理装置アプリケーション (LUA) インターフェースを使ってシステム・ネットワーク体系 (SNA) 準拠のホスト・コンピューターと通信する C 言語アプリケーション・プログラムを開発するためのマニュアルです。Communications Server for AIX がインプリメントしている LUA は、IBM® の OS/2® 製品 (Communications Server for OS/2 など) としてインプリメントしている要求/応答単位インターフェース (RUI) を基にして、AIX または Linux 環境に合わせた修正が加えてあります。

IBM Communications Server for AIX (以降、CS/AIX と呼ぶ) は、AIX® を使用しているサーバーが SNA ネットワーク上の他のノードと情報を交換するための IBM ソフトウェア製品です。

本書は、AIX バージョン 5.2 以降の基本オペレーティング・システム上で稼働する CS/AIX バージョン 6.3 に適用されます。

本書「*Communications Server for AIX LUA プログラマーズ・ガイド*」に関するご意見、ご要望は、中表紙裏に記載されている宛先にお送りください。

本書の対象読者

本書は、CS/AIX を備えたシステム用のシステム・ネットワーク体系 (SNA) トランザクション・プログラムを作成する、熟練した C 言語プログラマーを対象としています。ただし、SNA や CS/AIX の通信機能の実務経験はなくても構いません。

アプリケーション・プログラマーは、CS/AIX のプログラミング・インターフェースを使用して SNA ネットワークを介したデータの送受信を行うトランザクション・プログラムとアプリケーション・プログラムの、設計およびコーディングを行います。このため、SNA、トランザクション・プログラムまたはアプリケーション・プログラムの通信相手となるリモート・プログラム、および AIX または Linux オペレーティング・システムのプログラミング環境と操作環境に精通している必要があります。

アプリケーション・プログラム作成の詳細については、個々の API の解説書を参照してください。CS/AIX の資料の詳細については、参考文献を参照してください。

本書の使用法

この節では、本書の構成と表記上の規則を説明します。

本書の構成

本書は、次のように構成されています。

- 1 ページの『第 1 章 概念』では、LUA の基本的な概念を説明します。この章は、LUA に精通していないプログラマーを対象としています。

- 13 ページの『第 2 章 LUA アプリケーションの設計と作成』では、LUA アプリケーションを作成するときにプログラマーが必要とする一般的な情報を示します。この章には、LUA アプリケーションの設計に関連した SNA の概念や、LUA アプリケーションのコンパイルとリンクに関する情報も含まれています。
- 51 ページの『第 3 章 LUA VCB 構造体』では、すべての LUA verb に使用される verb 制御ブロック (verb control block: VCB) の構造を説明します。
- 63 ページの『第 4 章 RUI verb』では、各 RUI verb について詳しく説明します。説明する項目は、目的、verb のレコード形式、指定パラメーターと戻り値、その verb と他の RUI verb との関連などです。
- 115 ページの『第 5 章 SLI verb』では、各 SLI verb について詳しく説明します。説明する項目は、目的、verb のレコード形式、指定パラメーターと戻り値、その verb と他の SLI verb との関連などです。
- 171 ページの『第 6 章 LUA アプリケーションのサンプル』では、LUA RUI verb の使用法を示す CS/AIX LUA のサンプル・アプリケーションについて説明します。この章では、サンプル・アプリケーションのコンパイル、リンク、および実行の手順も説明します (必要な CS/AIX の構成手順も含まれます)。
- 177 ページの『付録 A. 戻りコードの値』では、LUA インターフェースの可能な戻りコードをすべて番号順にリストし、それぞれの意味を示します。

表記上の規則

表 1 は、本書で使用する表記上の規則です。

表 1. 表記上の規則

内容	表記例
語句の強調	ファイルを削除する前にバックアップしてください
資料名	「 <i>Communications Server for AIX</i> 管理者用ガイド」
ファイル名またはパス名	/usr/spool/uucp/myfile.bkp
プログラムまたはアプリケーション	snaadmin
コマンドまたは AIX/Linux ユーティリティ	define_node; cd
特定のタイプのコマンドすべてをまとめて参照する	query_* (リソースの詳細を照会するすべての管理コマンドを指します)
オプションまたはフラグ	-i
パラメーターまたは Motif フィールド	<i>opcode; LU name</i>
ユーザーが入力できるリテラル値または選択項目 (デフォルト値を含む)	255; On node startup
定数またはシグナル	AP_GET_LU_STATUS
戻り値	AP_INVALID_FORMAT; 0; -1
指定値を表す変数	<i>filename; LU_name; user_ID</i>
環境変数	PATH
プログラミング verb	GET_LU_STATUS
ユーザーの入力	Op1
コンピューターの実出力	CLOSE
関数、呼び出し、またはエントリー・ポイント	ioctl
データ構造体	termios
3270 のキー	ENTER
キーボードのキー	Ctrl+D; Enter

表 1. 表記上の規則 (続き)

内容	表記例
16 進値	0x20

シンボルの意味

AIX, LINUX

このシンボルは、AIX または Linux オペレーティング・システムにのみ適用されるテキストのセクションの開始を示すために使用されます。これは、AIX サーバーと、AIX 上で実行する IBM Remote API Client、Linux、Linux for pSeries、または Linux for zSeries に適用されます。

WINDOWS

このシンボルは、Windows 上の IBM Remote API Client に適用されるテキストのセクションの開始を示すために使用されます。

■

このシンボルは、オペレーティング・システム固有のテキストの終了を示すために使用されます。このシンボルに続く情報は、オペレーティング・システムを問わず、適用されます。

新機能

Communications Server for AIX V6.3 が Communications Server for AIX V6.1 に取って代わります。

サポートが続く本製品のリリースは、次のとおりです。

- Communications Server for AIX V6.1

サポートされなくなった本製品のリリースは、次のとおりです。

- Communications Server for AIX バージョン 6 (V6)
- Communications Server for AIX バージョン 5 (V5)
- Communications Server for AIX バージョン 4.2 (V4R2)
- Communications Server for AIX バージョン 4.1 (V4R1)
- SNA Server for AIX バージョン 3.1.1 (V3R1.1)
- SNA Server for AIX バージョン 3.1 (V3R1)
- AIX SNA Server/6000 バージョン 2.2 (V2R2)
- AIX 3.2 で稼働する AIX SNA Server/6000 バージョン 2.1 (V2R1)
- AIX SNA サービス/6000 バージョン 1

新機能

さらに、本書では、以下の変更が行われました。

- セッション・レベル・インターフェース (SLI) は、LUA インターフェースに組み込まれました。
- Communications Server for AIX は、LUA インターフェースの一部として 1 次 RUI をサポートするようになりました。これにより、ダウンストリーム PU との通信用の 1 次 SNA として動作するアプリケーションを作成することができます。

詳細について

CS/AIX ライブラリー内の他の資料や、SNA および AIX ワークステーションの関連事項について詳しい情報を記載した資料については、参考文献を参照してください。

CS/AIX 資料に収録されている情報は、HTML 形式でも入手できます。このライブラリーを使用して、特定の情報を検索したり、各 CS/AIX 資料のオンライン・バージョンを表示することができます。

第 1 章 概念

この章では、LUA (従来型の LU (論理装置) アプリケーション・プログラミング・インターフェース (API)) の基本概念を説明します。

この章では、次のトピックを扱います。

- LUA とは
- 使用するインターフェースの選択 (RUI または SLI)
- LU とセッション
- LUA verb
- LUA 通信順序の例
- LUA の互換性

LUA とは

LUA (従来型の LU アプリケーション・プログラミング・インターフェース) は、ホスト・アプリケーションと通信する CS/AIX アプリケーションを作成できるようにする API です。

LUA インターフェースは、要求/応答単位 (RU) レベルで提供されているため、プログラマーは、CS/AIX とホスト間で送信されるシステム・ネットワーク体系 (SNA) のメッセージを制御することができます。このインターフェースは、ホストの LU のタイプが、0、1、2、または 3 であればどのタイプとの通信にも使用できます。ホスト・アプリケーションが必要とする適切な SNA メッセージを送信するのは、アプリケーションの役割です。

たとえば、LUA を使用して、ホスト 3270 アプリケーションと通信する 3270 エミュレーション・プログラムを作成できます。この簡単な例がサンプル LUA アプリケーションとして CS/AIX に含まれています。このサンプル・プログラムについては 171 ページの『第 6 章 LUA アプリケーションのサンプル』で説明しています。

AIX, LINUX

ご使用の CS/AIX システムがダウンストリーム PU との通信用の SNA ゲートウェイをサポートしている場合、それらのダウンストリーム PU 上で 2 次 LU と通信するための 1 次 SNA として機能する LUA アプリケーションも作成できます。これにより、CS/AIX ノード上でホスト・アプリケーションをエミュレートしたり、ホスト・アプリケーションから CS/AIX ノードに処理をオフロードしたりすることができます。この機能を「1 次 RUI」と表現しています。これは、CS/AIX に固有のものであるため、他の LUA インプリメンテーションでは提供されないことがあります。



使用するインターフェースの選択

LUA には、異なるレベルに 2 つの異なるプログラミング・インターフェースが組み込まれています。

- 要求単位インターフェース (Request Unit Interface: RUI) は、要求/応答単位 (RU) レベルで提供されるため、プログラマーは、CS/AIX とホスト間で送信されるシステム・ネットワーク体系 (SNA) のメッセージを制御することができます。ホスト・アプリケーションが必要とする適切な SNA メッセージを作成および送信するのは、アプリケーションの役割です。

RUI インターフェースは SNA 機能管理プロファイル 2、3、4、7、18 および SNA 伝送サービス・プロファイル 2、3、4、7 をサポートします。

- セッション・レベル・インターフェース (SLI) は高水準インターフェースであるため、プログラマーは、個々の RU の詳細を気にすることなく、論理メッセージ・レベルで作業することができます。たとえば、次のとおりです。
 - セッションは、単一の SLI verb で確立したり、終了したりできます (セッションの起動および終了に関連する個々の RU に対応する一連の RUI verb を使用しなくてかまいません)。
 - アプリケーションが、BIND に指定された RU の最大長を超えるデータを送受信する必要がある場合、SLI ライブラリーがチェーニングを制御します。
 - ホストに送信されるほとんどの SNA コマンドについて、SLI ライブラリーは、アプリケーションの要求に応じて適切な RU を作成できます。

SLI インターフェースは、SNA 機能管理プロファイル 3 および 4 と SNA 伝送サービス・プロファイル 3 および 4 をサポートします。

アプリケーションは、それぞれのセッションごとに、これらのインターフェースのいずれか 1 つのみを使用できます。たとえば、RUI を使用してセッションを開始した場合、その後、そのセッションで SLI verb を発行することはできません。

使用する API を決定する前に、次の点を検討する必要があります。

- SLI は、個々の RU の詳細とその内容の一部を処理し、アプリケーションで必要な処理を単純化します。RUI では、アプリケーションが、それぞれの RU を個別に扱う必要があります。
- RUI は、ホストに送信される RU の詳細な内容まで制御できるので、広範な SNA バインド・プロファイルを使用できます。SLI には、同じ程度の制御や柔軟性は備わっていません。

AIX、LINUX

- RUI には 1 次 RUI が組み込まれています (AIX または Linux のみ)。これにより、ダウンストリーム PU と通信するための 1 次 SNA として動作するアプリケーションを作成することができます。SLI インターフェースでは、この機能は提供されません。

LU とセッション

図 1 は、ホストとの LUA 通信に使用される SNA コンポーネントを示したものです。

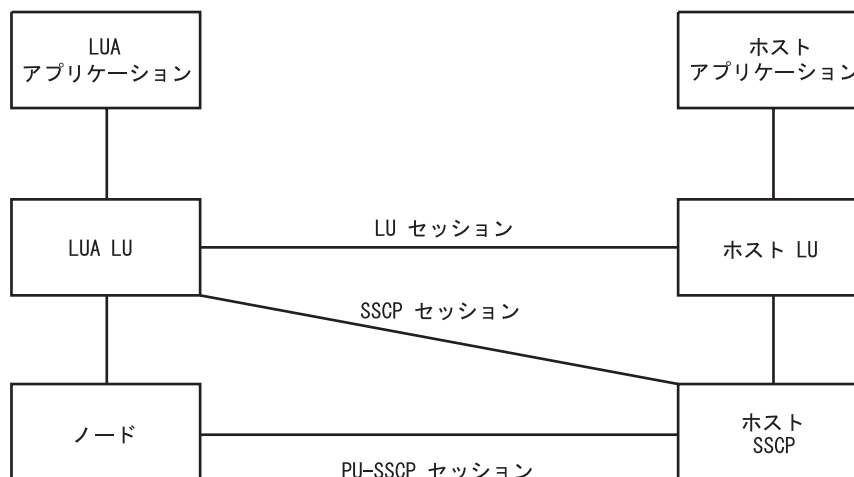


図 1. LUA 通信に使用される SNA コンポーネント

LUA アプリケーションは、CS/AIX ノードを使用してホスト・システムと通信する、タイプ 0 から 3 の LU を使用します。CS/AIX ノードとホスト・ノードの間には、次の 3 つのセッションがあります。

- 物理装置 - システム・サービス制御点 (PU-SSCP) セッション。これは PU 2.1 とホストのシステム・サービス制御点 (SSCP) 間のセッションで、PU の制御に使用されます。
- CS/AIX LU と SSCP 間の SSCP セッション。LU の制御に使用されます。
- CS/AIX LU とホスト LU 間の LU セッション。LU とホスト・アプリケーションの間のデータ転送に使用されます。

LUA アプリケーション・プログラミング・インターフェースを使用すると、アプリケーションは SSCP セッションと LU セッションでデータを送受信できます。ただし、このインターフェースは、PU-SSCP セッションへのアクセスは提供しません。LUA アプリケーションが PU-SSCP セッションでデータを送信するには、管理サービス (MS) verb TRANSFER_MS_DATA を使用します。詳しくは、「*Communications Server for AIX MS プログラマーズ・ガイド*」を参照してください。

WINDOWS

Windows オペレーティング・システムでは、TRANSFER_MS_DATA は、Common Service Verb (CSV) API の一部として提供されます。詳しくは、「*Communications Server for AIX CSV プログラマーズ・ガイド*」を参照してください。

AIX、LINUX

図2 は、ダウンストリーム LU に対する 1 次 RUI を使用した LUA 通信に使用される SNA コンポーネントを示したものです。

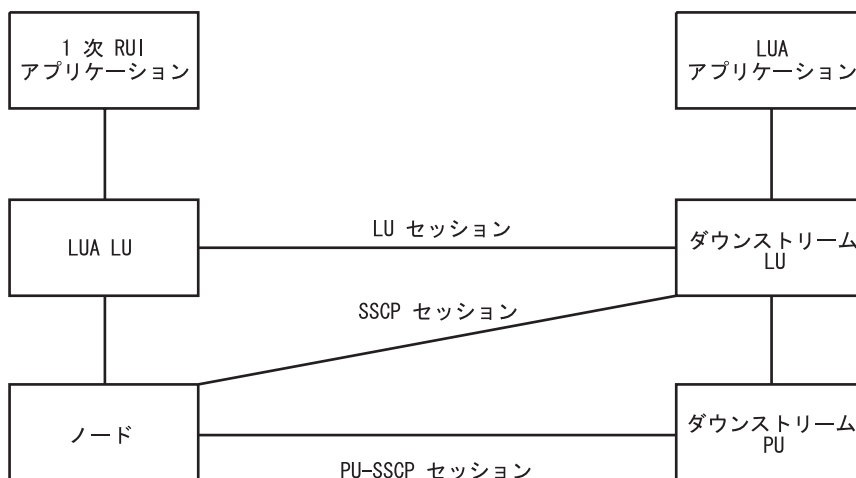


図2. 1 次 RUI 通信に使用される SNA コンポーネント

1 次 RUI アプリケーションは、CS/AIX ノードを使用してダウンストリーム LU と通信する、タイプ 0 から 3 の LU を使用します。ダウンストリーム LU から見ると、CS/AIX LU はホスト LU として機能し、CS/AIX ノードはホスト SSCP として機能します。これらのコンポーネント間の 3 つのセッションと、これらのセッションへのアクセスに対する制約事項は、ホストと通信する LUA アプリケーションの場合と同じです。

個々の LU セッションでは、メッセージの優先順位として通常と急送の 2 つがあります。急送フロー・メッセージは、同じセッションで送信を待っている他のメッセージよりも優先されます。メッセージの送受信には、次の 4 つのフローを使用できます。

- SSCP セッション、急送フロー
- LU セッション、急送フロー
- SSCP セッション、通常フロー
- LU セッション、通常フロー

LU セッションの通常フローでは、アプリケーション・データが送受信されます。他のフローは、制御メッセージと始動に使用されます。

CS/AIX にインプリメントされた LUA では、アプリケーションは SSCP 急送フローでデータを送信することはできず、このフローではデータはアプリケーションに戻りません。

構成

LUA が使用する個々の LU は、Motif 管理プログラム、コマンド行管理プログラム、あるいは Node Operator Facility (NOF) API を使用して構成する必要があります (詳しくは、「*Communications Server for AIX* 管理ガイド」または「*Communications Server for AIX NOF プログラマーズ・ガイド*」を参照してください)。さらに、CS/AIX の構成には、LU プールが組み込まれることがあります。プールとは、類似した特性をもつ LU のグループであり、アプリケーションはグループ内の空いている LU をどれでも使用できます。使用可能な LU の数よりアプリケーションの数の方が多く場合、プールを使うと、LU を先着順に割り振ることができます。また、プールを使用することにより、接続ごとに LU を選択することができます。

LUA verb

アプリケーションは、LUA verb を使って LUA にアクセスします。各 verb は LUA にパラメーターを提供し、LUA は該当する関数を実行してアプリケーションにパラメーターを戻します。

RUI verb の概要

次に示すリストは、各 LUA RUI verb の概要を示したものです (各 verb の詳細については、63 ページの『第 4 章 RUI verb』を参照してください)。

RUI_BID

この verb は、ホストからの情報の読み取りが可能になったときに、アプリケーションがそのことを判別できるようにします。

RUI_INIT

この verb は、LUA アプリケーション用の SSCP セッションをセットアップします。

AIX, LINUX

RUI_INIT_PRIMARY

この verb は、ダウンストリーム LU と通信するための 1 次 SNA として機能する LUA アプリケーション用の SSCP セッションをセットアップします。

RUI_PURGE

この verb は、未解決の RUI_READ verb を取り消します。

RUI_READ

この verb は、SSCP セッションまたは LU セッションで、ホストから LUA アプリケーションの LU に送信されたデータまたは状況情報を受け取ります。

AIX, LINUX

RUI_REINIT

この verb は、セッションに障害が発生した後で、LUA アプリケーション用の SSCP セッションを再確立します。この verb は、プールの LU を使用していたアプリケーションが、処理を続行するために 同じ LU を使用し、セッションを再確立する必要がある場合に使用するためのものです。

RUI_TERM

この verb は、LUA アプリケーション用の SSCP セッションを終了します。LU セッションがアクティブである場合は、LU セッションも終了します。

RUI_WRITE

この verb は、SSCP セッションまたは LU セッションでホストにデータを送信します。

SLI verb の概要

次に示すリストは、各 LUA SLI verb の概要を示したものです (各 verb の詳細については、115 ページの『第 5 章 SLI verb』を参照してください)。

SLI_BID

この verb は、ホストからの情報の読み取りが可能になったときに、アプリケーションがそのことを判別できるようにします。

SLI_CLOSE

この verb は、LUA アプリケーション用のセッションを終了します。

SLI_OPEN

この verb は、LUA アプリケーション用のセッションをセットアップします。

SLI_PURGE

この verb は、未解決の SLI_RECEIVE verb を取り消します。

SLI_RECEIVE

この verb は、SSCP セッションまたは LU セッションで、ホストから LUA アプリケーションの LU に送信されたデータまたは状況情報を受け取ります。

SLI_SEND

この verb は、SSCP セッションまたは LU セッションでホストにデータを送信します。

アプリケーションは、ホストからの BIND、STSN、および SDT 要求を処理するために、独自のルーチンのアドレスをオプションで SLI_OPEN verb に指定できます。アプリケーションがこのようなルーチンを提供している場合、該当するタイプの要求がホストから到着すると、LUA は次のように、アプリケーションが提供する適切なルーチンに追加の verb を送信して、そのルーチンが要求を処理できるようにします。

SLI_BIND_ROUTINE

ホストから BIND 要求が到着すると、LUA は、アプリケーションが提供する BIND ルーチンにこの verb を送信します。アプリケーションは、38 ページの『SNA 情報』に記されているように、この BIND を受け入れるか、BIND パラメーターについて折衝する、あるいは BIND をリジェクトします。

アプリケーションが BIND ルーチンを提供しない場合、LUA は、限定された BIND 検査を行い、ホストに適宜応答します。

SLI_STSN_ROUTINE

ホストから STSN 要求が到着すると、LUA は、アプリケーションが提供する STSN ルーチンにこの verb を送信します。アプリケーションは、38 ページの『SNA 情報』に記されているように、STSN に応答するか、あるいは適切な SNA センス・コードを使用してリジェクトすることができます。

アプリケーションが STSN ルーチンを提供しない場合、LUA は、使用可能なデータがないことを示す肯定応答を戻します。

SLI_SDT_ROUTINE

ホストから SDT 要求が到着すると、LUA は、アプリケーションが提供する SDT ルーチンにこの verb を送信します。38 ページの『SNA 情報』に記されているように、アプリケーションは、SDT に応答するか、あるいは適切な SNA センス・コードを使用してリジェクトできます。

アプリケーションが SDT ルーチンを提供しない場合、LUA は、肯定応答を戻します。

verb の非同期完了

LUA verb の中には、ローカル処理が終わるとただちに完了するものもあります (RUI_PURGE verb など)。しかし、大部分の verb は、完了するのにしばらく時間がかかります。これは、ノードやホストとの間でメッセージをやり取りする必要があるためです。このため、LUA は非同期インターフェースとしてインプリメントされています。verb がまだ進行中でもアプリケーションに制御を戻すことができるので、アプリケーションは処理を続行できます (他の LUA verb を発行することもできます)。

AIX、LINUX

verb が完了すると、LUA はアプリケーションが提供するコールバック・ルーチンを呼び出します。コールバック・ルーチンには、戻されたデータをさらに処理するもの、さらに LUA verb を発行するもの、あるいは verb が完了したことを示すインディケータの役割を果たすだけのものがあります。

- RUI verb は同期して完了すること、非同期で完了することもあります。アプリケーションで、VCB 内の 1 次戻りコードを検査し、それぞれの verb に適用される完了コードを判別する必要があります。
- SLI verbs は常に非同期で完了します。アプリケーションは、verb を発行したら、コールバック・ルーチンが呼び出されるまで VCB をアクセスできません。VCB は、コールバック・ルーチンから、あるいはコールバック・ルーチンの完了後は、プログラムの実行のメイン・スレッドから処理できます。

WINDOWS

verb が完了すると、LUA は、アプリケーションが提供するウィンドウ・ハンドルへのメッセージを通知するか、アプリケーションが提供するイベント・ハンドルにシグナルを送ります。

詳しくは、13 ページの『第 2 章 LUA アプリケーションの設計と作成』を参照してください。

LUA 通信順序の例

10 ページの図 3 には、RUI verb を使用した LUA 通信順序の例を示し、11 ページの図 4 には、SLI verb を使用した場合の同じ通信順序を示します。

RUI の例では、アプリケーションは次の処理を実行します。

1. RUI_INIT verb を発行して SSCP セッションを確立します。RUI_INIT verb は、CS/AIX が論理装置活動化 (ACTLU) メッセージをホストから受け取り、肯定応答を送信するまで完了しません。ただし、これらのメッセージは CS/AIX が処理し、LUA アプリケーションには公開されません。
2. INITSELF メッセージを SSCP に送信して BIND を要求し、その応答を読み取ります。
3. ホストからの BIND メッセージを読み取って、応答を書き込みます。これによって、LU セッションが確立されます。
4. ホストからの SDT メッセージを読み取ります。このメッセージには、初期化が完了し、データ転送を開始できる状態になったことが示されています。
5. 3 つの RU (最後の RU は、確定応答が必要なことを示します) から成るデータのチェーンを送信し、応答を読み取ります。
6. 2 つの RU から成るデータのチェーンを読み取って、応答を書き込みます。
7. ホストからの UNBIND メッセージを読み取って、応答を書き込みます。これによって、LU セッションが終了します。
8. RUI_TERM verb を発行して SSCP セッションを終了します (CS/AIX は NOTIFY メッセージをホストに送信し、肯定応答を待ちます。ただし、これらのメッセージは CS/AIX で処理され、LUA アプリケーションには公開されません)。

SLI の例は、ホストとアプリケーション間の、同じ順序のメッセージ・フローを示しています。使用されている SLI verb は、RUI の例で使用されているものと同じですが、以下の違いがあることに注意してください。

- SLI_OPEN は、セッションの完全初期化を処理する。RUI の例のように、アプリケーションが、初期化シーケンスに含まれる個々の RU を読み書きする必要はありません。
- LUA は、アプリケーションの BIND ルーチンおよび SDT ルーチン (SLI_OPEN に指定される) を使用して、アプリケーションがホストからの BIND メッセージおよび SDT メッセージを処理できるようにする。これらのルーチンは同期して戻る必要があります。その他の SLI verb はすべて非同期で完了します。
- SLI_RECEIVE ハンドルおよび SLI_SEND ハンドルがデータのチェーン全体を処理するため、2、3 個の RU が必要なほどの長さのデータであっても、その送受信にアプリケーションが必要とする verb は 1 つだけである (RUI の例では、アプリケーションが、各 RU を送受信するためには、別個の verb を使用する必要があります)。

次に、10 ページの図 3 および 11 ページの図 4 で使用されている省略語の一覧を示します。

SSCP norm	SSCP セッション、通常フロー
LU norm	LU セッション、通常フロー
LU exp	LU セッション、急送フロー
+rsp	指示メッセージに対する肯定応答
BC	チェーン開始
MC	チェーンの途中
EC	チェーン終了
CD	方向転換インディケータのセット
RQD	確定応答が必要

10 ページの図 3 に、セッションの開始、データの交換、およびセッションの終了に使用する RUI verb と送受信される SNA メッセージを示します。矢印は、SNA メッセージが送られる方向を示します。

例、LUA 通信順序の

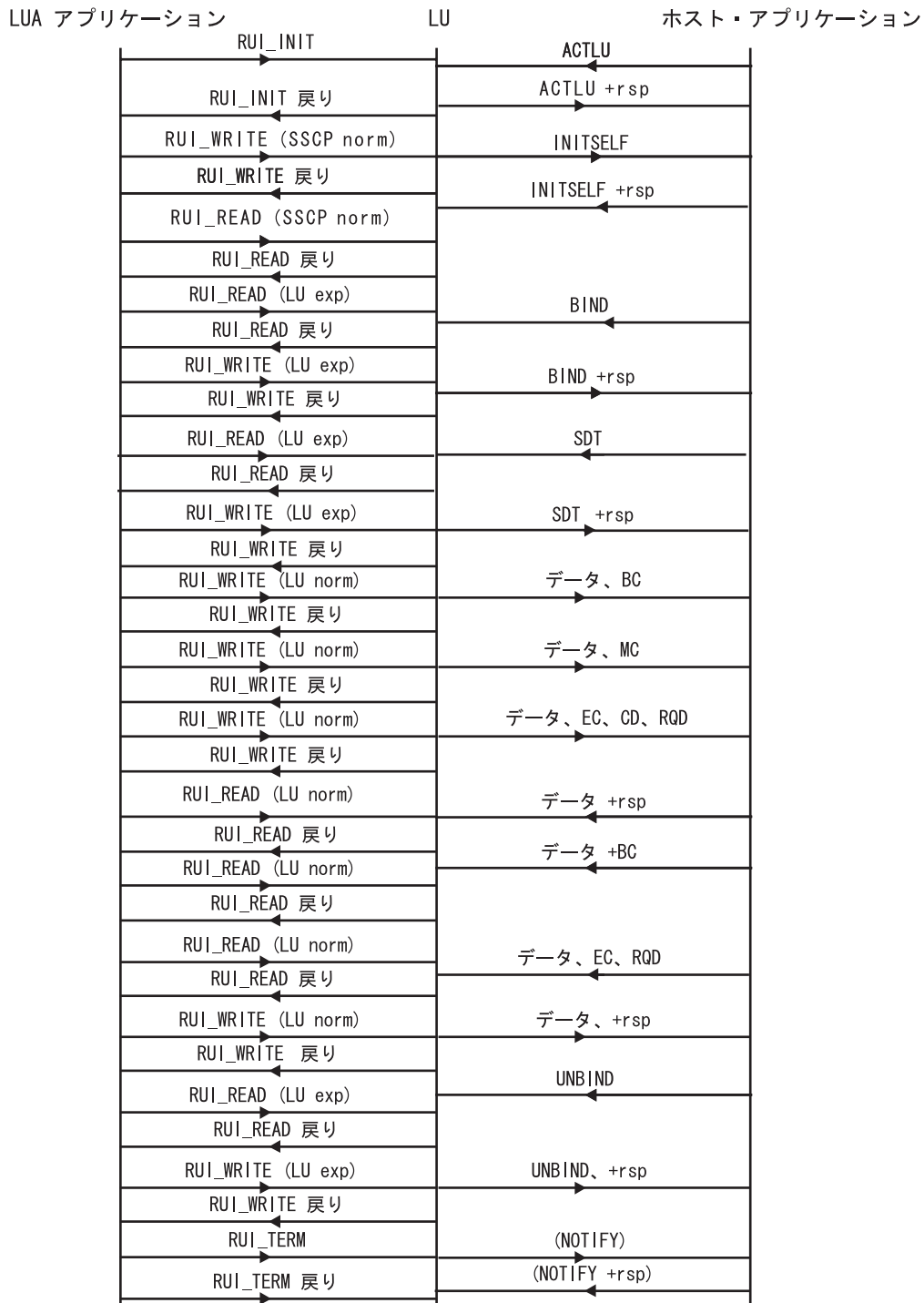


図3. RUI 通信順序

11 ページの図4 は、同じ SNA メッセージ・シーケンスに使用される同等な SLI verb を示しています。

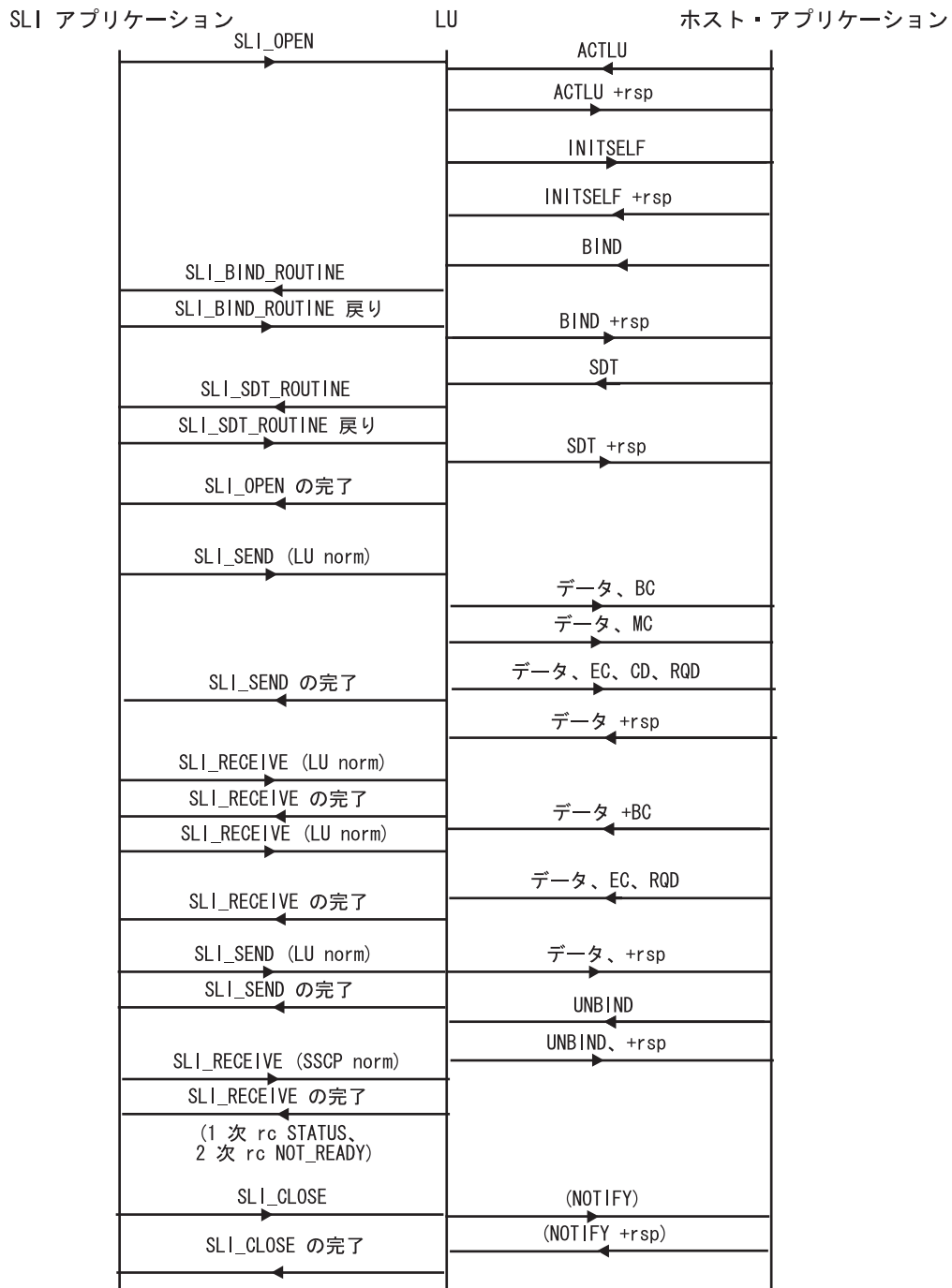


図 4. SLI 通信順序

LUA の互換性

AIX、LINUX

LUA の互換性

RUI_INIT_PRIMARY verb および RUI_REINIT verb は、標準 LUA インターフェース仕様に対する拡張機能です。これらは、Windows 上の Remote API Client では使用できないため、他の LUA インプリメンテーションでも使用できない場合があります。

WINDOWS

Windows 上の Remote API Client 上にインプリメントされた LUA は、Windows LUA と互換性をもつように設計されており (WOSA SNA 仕様により定義されています)、Windows LUA 用に作成されたアプリケーションは、変更なしで、Remote API Client で使用できます。



第 2 章 LUA アプリケーションの設計と作成

この章では、LUA アプリケーション・プログラムの作成に役立つ情報を提供します。この章では、次のトピックを扱います。

- AIX または Linux アプリケーションの LUA エントリー・ポイント
- Windows アプリケーションの LUA エントリー・ポイント
- LUA verb の発行
- SNA 情報
- 構成情報
- AIX または Linux に関する考慮事項
- Windows に関する考慮事項
- 移植可能なアプリケーションの作成

AIX または Linux アプリケーションの LUA エントリー・ポイント

AIX または Linux 上で実行するアプリケーションは、LUA verb に関する情報を含む Verb 制御ブロック (VCB) のアドレスを指定し、RUI 関数呼び出しまたは SLI 関数呼び出しを使用して、LUA にアクセスします。CS/AIX は、即時にアプリケーションに制御を戻します。

戻された VCB には、verb の処理がまだ進行中であるのか、完了したのかを示す値が入っています。

- 制御がアプリケーションに戻されても verb 処理が進行中の場合もあります。この場合、CS/AIX は、アプリケーションが提供するコールバック・ルーチンを使って verb 処理の結果を戻します。
- 一方、verb 処理が CS/AIX が制御をアプリケーションに返す時点で完了している場合は、CS/AIX はアプリケーションのコールバック・ルーチンを使用しません。特に、verb が LUA の初期パラメーター検査または状態検査に不合格であった場合がこれに該当し、verb は実行できません。
- SLI_OPEN の場合、初期検査に合格すると、SLI 関数呼び出しから新しいセッションのセッション ID を表す非ゼロ値が戻されます。次いで、CS/AIX は、アプリケーションが提供するコールバック・ルーチンを他の verb と同様に使用します。アプリケーションは、新しいセッション ID を使用することによって、コールバック・ルーチンが呼び出されるのを待たずに、セッションの後続の verb のうち限定された範囲のものを発行できます。このような状態で発行できる verb の詳細については、139 ページの『他の verb との関連』を参照してください。

注: オペレーティング・システム・コールバック・ルーチンの作動の仕組みにより、verb に対する初期関数呼び出しからアプリケーションへ制御が戻る前にアプリケーションのコールバック・ルーチンが呼び出されるようにすることができます。つまり、コールバック・ルーチンが、戻された VCB を変更または削除した場合、プログラムの実行のメイン・スレッドは、その verb が非同期で作

AIX または Linux アプリケーションの LUA エントリー・ポイント

動しているかどうかを判別するために VCB パラメーターを検査できないことがあります。アプリケーション設計では、このことを考慮に入れる必要があります。

エントリー・ポイント RUI および SLI は、LUA ヘッダー・ファイル `/usr/include/sna/luac.h` (AIX) または `/opt/ibm/sna/include/luac.h` (Linux) で定義されます。

RUI 関数呼び出し

```
void RUI(verb)
LUA_VERB_RECORD * verb;
```

SLI 関数呼び出し

```
AP_UINT32 SLI(verb)
LUA_VERB_RECORD * verb;
```

指定パラメーター

指定パラメーターは次のとおりです。

verb 発行される *verb* のパラメーターが入っている *verb* 制御ブロック (VCB) を指すポインター。VCB 構造体は、LUA ヘッダー・ファイル `luac.h` で定義されています。詳しくは、51 ページの『第 3 章 LUA VCB 構造体』を参照してください。

注: LUA VCB には、「予約済み」としてマークされた多くのパラメーターが含まれています。これらの予約済みパラメーターには、CS/AIX ソフトウェアにより内部的に使用されるものがありますが、このバージョンでは使用されなくても将来のバージョンで使用されるものもあります。アプリケーションでは、これらの予約済みパラメーターには決してアクセスしないでください。*verb* によって使用される他のパラメーターをアプリケーションが設定する前に、VCB の内容全体をゼロに設定して、これらのパラメーターすべてを確実にゼロに設定しておく必要があります。このように設定しておくと、CS/AIX がその内部使用パラメーターを誤って解釈することはありません。またこれにより、今後の CS/AIX のバージョンで、これらのパラメーターを使って新しい関数を引き続き使用できるようになります。

VCB の内容をゼロに設定するには、次のように `memset` を使用します。

```
memset(vcb, 0, sizeof(vcb));
```

戻り値

RUI の *verb*、および SLI_OPEN 以外のすべての SLI の *verb* の場合、エントリー・ポイントは値を戻しません。VCB 内に戻される値は、*verb* が同期して完了したのか、非同期で完了するのを示します。*verb* の完了後、VCB にはその *verb* の結果が含まれます。

SLI_OPEN の場合、エントリー・ポイントは、VCB が LUA の初期検査に合格したかどうかを示す値を戻します。

- 戻り値 0 (ゼロ) は、verb が LUA の初期検査に不合格であったことを示します (アプリケーションが提供したパラメーターが正しくなかった場合など)。CS/AIX では、アプリケーションが提供するコールバック・ルーチンは呼び出されません。
- ゼロ以外の値は、SLI_OPEN によって開始される新規セッションのセッション ID を表します。

この戻り値は、verb が完了したことを示しているわけではありません。CS/AIX は、セッションがセットアップされると、アプリケーションが提供するコールバック・ルーチンを呼び出して、SLI_OPEN の完了を示します。

詳しくは、『使用法』を参照してください。

使用法

verb が発行されるとすぐに、LUA がその verb の処理をすべて完了できる場合があります。特に、verb が LUA の初期パラメーター検査または状態検査に不合格であった場合がこれに該当し、verb は実行できません。この場合、verb は同期復帰します。1 次戻りコードは LUA_IN_PROGRESS 以外の値に設定され、*lua_flag2.async* ビットは 0 (ゼロ) に設定されます (これらの戻りパラメーターについては、63 ページの『第 4 章 RUI verb』または 115 ページの『第 5 章 SLI verb』を参照してください)。

また一方では、LUA は verb を完了する前に、リモート LU またはノードからの情報を待つ必要がある場合もあります。この場合、verb は非同期復帰します。1 次戻りコードは LUA_IN_PROGRESS に設定され、*lua_flag2.async* ビットは 1 に設定されます。アプリケーションは、この時点で別の処理を実行することもできますし、verb が完了したという LUA からの通知を待つこともできます。LUA は、1 次戻りコードをその最終値に設定し、*lua_flag2.async* は 1 に設定されたままにして、この通知を発行します。

アプリケーションは、提供する VCB の一部として、コールバック・ルーチンを指すポインタを (*lua_post_handle* パラメーターで) 提供します。verb が同期して完了した場合、LUA はコールバック・ルーチンを呼び出しません。verb が非同期で完了した場合、LUA は 1 つのパラメーター (オリジナルの verb 制御ブロック (VCB) を指すポインタ) を指定してコールバック・ルーチンを呼び出すことにより、verb の完了を示します。詳しくは、16 ページの『verb の非同期完了時用のコールバック・ルーチン』を参照してください。

注:

1. ある verb が同期して完了するのか、それとも非同期で完了するのかを、アプリケーション側で予測することはできません。
2. *lua_flag2.async* パラメーターが、verb が非同期で完了することを示している場合、その時点で VCB 内にある他のパラメーターにプログラムの実行のメイン・スレッドがアクセスすることはできません。LUA がコールバック・ルーチンを呼び出すと、アプリケーションは VCB パラメーターにアクセスできるようになります。

- オペレーティング・システム・コールバック・ルーチンの作動の仕組みにより、`verb` に対する初期関数呼び出しからアプリケーションへ制御が戻る前にアプリケーションのコールバック・ルーチンが呼び出されるようにすることができます。つまり、コールバック・ルーチンが、戻された VCB を変更または削除した場合、プログラムの実行のメイン・スレッドは、その `verb` が非同期で作動しているかどうかを判別するために VCB パラメーターを検査できないことがあります。アプリケーション設計では、このことを考慮に入れる必要があります。

verb の非同期完了時用のコールバック・ルーチン

LUA `verb` を非同期で完了できるようにするには、コールバック・ルーチンを指すポインターをアプリケーションで指定する必要があります。この節では、CS/AIX がこのルーチンをどのように使用するかについて説明し、CS/AIX が実行する必要のある関数について説明します。

関数呼び出し

```
void callback (verb)
LUA_VERB_RECORD * verb;
{
    .
    .
}
```

指定パラメーター

CS/AIX は、次のパラメーターを指定してこのルーチン呼び出します。

verb アプリケーションによって提供された VCB を指すポインター (この中には CS/AIX が設定した戻りパラメーターが入っています)。コールバック・ルーチンは、VCB 内の戻りパラメーターに対して必要なすべての処理を実行する場合も、`verb` が完了したことをメインプログラムに通知する変数を設定するだけの場合もあります。

注: オペレーティング・システム・コールバック・ルーチンの作動の方法により、`verb` に対する初期関数呼び出しからアプリケーションへ制御が戻る前にアプリケーションのコールバック・ルーチンが呼び出されるようにできます。つまり、コールバック・ルーチンが、戻された VCB を変更または削除した場合、プログラムの実行のメイン・スレッドは、その `verb` が非同期で作動しているかどうかを判別するために VCB パラメーターを検査できないことがあります。アプリケーション設計では、このことを考慮に入れる必要があります。

戻り値

戻り値はありません。

Windows アプリケーションの LUA エントリー・ポイント

WINDOWS

Windows アプリケーションは、以下の関数を使用して LUA にアクセスします。

Windows アプリケーションの LUA エントリー・ポイント

RUI RUI verb を発行します。verb が非同期で完了すると、LUA は、アプリケーションが提供するイベント・ハンドルにシグナルを送ることによって完了を通知します。

WinRUIStartup

Windows RUI ユーザーとして登録し、LUA ソフトウェアが、アプリケーションに必要な関数レベルをサポートするかどうかを判別します。

WinRUI RUI verb を発行します。verb が非同期で完了した場合、LUA はアプリケーション・ウィンドウにメッセージを通知することによって完了を通知します。

WinRUIGetLastInitStatus

RUI セッション (未解決の直前の RUI_INIT verb によって開始された) の状況を検査して、セッション状況に対する変更の通知を要求するか、その通知を取り消します。

WinRUICleanup

アプリケーションが RUI の使用を終了したら、そのアプリケーションの登録を抹消します。

GetLuaReturnCode

LUA verb から得られた 1 次および 2 次戻りコードの印刷可能文字ストリングを生成します。

SLI SLI verb を発行します。verb が非同期で完了すると、LUA は、アプリケーションが提供するイベント・ハンドルにシグナルを送ることによって完了を通知します。

WinSLIStartup

Windows SLI ユーザーとして登録し、LUA ソフトウェアが、アプリケーションに必要な関数レベルをサポートするかどうかを判別します。

WinSLI SLI verb を発行します。verb が非同期で完了した場合、LUA はアプリケーション・ウィンドウにメッセージを通知することによって完了を通知します。

WinSLICleanup

アプリケーションが SLI の使用を終了したら、そのアプリケーションの登録を抹消します。

RUI アプリケーションは、WinRUI 呼び出しを使用して LUA verb を発行する前に、WinRUIStartup を呼び出す必要があります。

RUI_INIT verb が未解決である間、アプリケーションは、WinRUIGetLastInitStatus を使用して、その verb によって開始された LUA セッションの状況を判別できません。その結果、必要に応じて、RUI_INIT verb を取り消すことができます。

WinRUIGetLastInitStatus 関数を使用すると、以降の変更の通知を要求せずに現在の状況の検査、セッション状況に対する以降の変更の非同期通知の要求、または前に出された状況変更の通知要求の取り消しもできます。

verb から LUA_OK 以外の戻りコードが返された場合、アプリケーションは GetLuaReturnCode を使用して、それらの戻りコードのテキスト・ストリング表記を取得できます。これを使って、標準のエラー・メッセージを生成できます。

Windows アプリケーションの LUA エントリー・ポイント

WinRUI 呼び出しを使用して LUA verb の発行を終了したら、WinRUICleanup を呼び出してから終了する必要があります。WinRUICleanup を呼び出した後ではもう、RUI verb を発行しないようにしてください。

SLI アプリケーションは、WinSLI 呼び出しを使用して LUA verb を発行する前に、WinSLIStartup を呼び出す必要があります。

verb から LUA_OK 以外の戻りコードが返された場合、アプリケーションは GetLuaReturnCode を使用して、それらの戻りコードのテキスト・ストリング表記を取得できます。これを使って、標準のエラー・メッセージを生成できます。

WinSLI 呼び出しを使用して LUA verb の発行を終了したら、WinSLICleanup を呼び出してから終了する必要があります。WinSLICleanup を呼び出した後ではもう、SLI verb を発行しないようにしてください。

以下の節では、これらの関数を説明します。

RUI

アプリケーションは、この関数を使用して、LUA RUI verb を発行します。verb が非同期で完了すると、LUA は、アプリケーションが提供するイベント・ハンドルのシグナルを送ることによって完了を通知します。

この呼び出しを行う前に、アプリケーションで WinRUIStartup verb を発行する必要はありません。

関数呼び出し

```
void WINAPI RUI(verb)
LUA_VERB_RECORD FAR * verb;
```

指定パラメーター

指定パラメーターは次のとおりです。

verb 発行される verb のパラメーターが入っている verb 制御ブロック (VCB) を指すポインター。VCB 構造体は、LUA ヘッダー・ファイル **winlua.h** で定義されます。このファイルは、Windows クライアント・ソフトウェアのインストール先ディレクトリー内の、サブディレクトリー **/sdk** にインストールされます。VCB 構造体についての詳細は、51 ページの『第 3 章 LUA VCB 構造体』を参照してください。

注: LUA VCB には、「予約済み」としてマークされた多くのパラメーターが含まれています。これらの予約済みパラメーターには、CS/AIX ソフトウェアにより内部的に使用されるものがありますが、このバージョンでは使用されなくても将来のバージョンで使用されるものもあります。アプリケーションでは、これらの予約済みパラメーターには決してアクセスしないでください。verb によって使用される他のパラメーターをアプリケーションが設定する前に、VCB の内容全体をゼロに設定して、これらのパラメーターすべてを確実にゼロに設定しておく必要があります。このように設定しておくこと、CS/AIX がその内部使用パラメーター

Windows アプリケーションの LUA エントリー・ポイント

を誤って解釈することはありません。またこれにより、今後の CS/AIX のバージョンで、これらのパラメーターを使って新しい関数を引き続き使用できるようになります。

VCB の内容をゼロに設定するには、次のように `memset` を使用します。

```
memset(vcb, 0, sizeof(vcb));
```

戻り値

このエントリー・ポイントは値を戻しません。呼び出しから制御が戻されると、アプリケーションは VCB 内のパラメーターを調べて、その `verb` が同期して完了したのか、非同期で完了するのかを確認できます。詳しくは、『使用法』を参照してください。

使用法

`verb` が発行されるとすぐに、LUA がその `verb` の処理をすべて完了できる場合があります。この場合、`verb` は同期復帰します。1 次戻りコードは `LUA_IN_PROGRESS` 以外の値に設定され、`lua_flag2.async` ビットは 0 (ゼロ) に設定されます (これらの戻りパラメーターについては、63 ページの『第 4 章 RUI verb』を参照してください)。

また一方では、LUA は `verb` を完了する前に、リモート LU またはノードからの情報を待つ必要がある場合もあります。この場合、`verb` は非同期復帰します。1 次戻りコードは `LUA_IN_PROGRESS` に設定され、`lua_flag2.async` ビットは 1 に設定されます。アプリケーションは、この時点で別の処理を実行することもできますし、`verb` が完了したという LUA からの通知を待つこともできます。LUA は、1 次戻りコードをその最終値に設定し、`lua_flag2.async` ビットを 1 に設定したままで、この通知を発行します。

アプリケーションは、提供する VCB の一部として、`lua_post_handle` パラメーターでイベント・ハンドルを提供します。イベントは非シグナル状態で、ハンドルは、そのイベントに対する `EVENT_MODIFY_STATE` アクセスが可能でなければなりません。`verb` が同期して完了した場合は、LUA はそのイベント・ハンドルにシグナルを送りません。`verb` が非同期で完了すると、LUA は、イベント・ハンドルにシグナルを送ることによって完了を通知します。

アプリケーションは、イベント・ハンドルを待機するために `WaitForSingleObject` または `WaitForMultipleObject` 呼び出しを発行します。イベントがシグナル状態になると、アプリケーションは、1 次戻りコードと 2 次戻りコードのエラー検査を行います。

ある `verb` が同期して完了するのか、それとも非同期で完了するのかを、アプリケーション側で予測することはできません。

WinRUIStartup

アプリケーションはこの関数を使用して、Windows RUI ユーザーとして登録し、LUA ソフトウェアがアプリケーションに必要な Windows LUA バージョンをサポートしているかどうかを判別します。

関数呼び出し

```
int WINAPI WinRUIStartup (
    WORD wVersionRequired;
    LUADATA far * lpData;
)

typedef struct
{
    WORD wVersion;
    char szDescription[41];
} LUADATA;
```

指定パラメーター

指定パラメーターは次のとおりです。

wVersionRequired

アプリケーションに必要な Windows LUA のバージョン。CS/AIX は、バージョン 1.0 をサポートしています。

下位バイトには、メジャー・バージョン番号を指定し、高位バイトにはマイナー・バージョン番号を指定します。たとえば、次のように指定します。

バージョン	<i>wVersionRequired</i>
1.0	0x0001
1.1	0x0101
2.0	0x0002

アプリケーションが複数のバージョンを使用する場合、使用できる最高位のバージョンを指定する必要があります。

戻り値

関数の戻り値は、以下のいずれかです。

0 (ゼロ)

アプリケーションは正常に登録されており、Windows LUA ソフトウェアは、アプリケーションによって指定されたバージョン番号、あるいはその下位バージョンのどちらかをサポートしています。アプリケーションは、LUADATA 構造体内のバージョン番号を検査して、それが十分なレベルであることを確認する必要があります。

WLUAVERNOTSUPPORTED

アプリケーションで指定されたバージョン番号が、Windows LUA ソフトウェアでサポートされていません。アプリケーションは登録されませんでした。

WLUAINITREJECT

アプリケーションはすでに WinRUIStartup を呼び出しており、正常に登録されています。この関数を複数回呼び出すことはできません。

WLUASYSNOTREADY

CS/AIX ソフトウェアが開始されなかったか、ローカル・ノードがアクティブではありません。アプリケーションは登録されませんでした。

WLUAFailure

Windows LUA ソフトウェアの初期化中に、オペレーティング・システムで

Windows アプリケーションの LUA エントリー・ポイント

エラーが発生しました。アプリケーションは登録されませんでした。ログ・ファイルを確認して、エラーの原因を示すメッセージを参照してください。

WinRUIStartup からの戻り値が 0 (ゼロ) である場合、LUADATA 構造体に、Windows LUA ソフトウェアによって提供されるサポート情報が含まれています。戻り値がゼロ以外の値である場合、この構造体の内容は未定義で、アプリケーションは構造体の内容をチェックする必要はありません。この構造体にあるパラメーターは次のとおりです。

wVersion

ソフトウェアがサポートする Windows LUA バージョン番号で、*wVersionRequired* パラメーターと同じ形式です (20 ページの『指定パラメーター』を参照してください)。CS/AIX は、バージョン 1.0 をサポートしています。

ソフトウェアが要求されるバージョン番号をサポートする場合、このパラメーターは *wVersionRequired* パラメーターと同一の値に設定されます。そうでない場合、ソフトウェアがサポートする最高位のバージョンに設定されますが、これは、アプリケーションに指定されるバージョン番号より低位になります。アプリケーションは戻り値を検査し、以下のようなアクションを行います。

- 戻されたバージョン番号が要求されたバージョン番号と同一である場合、アプリケーションはこの Windows LUA インプリメンテーションを使用することができます。
- 戻されたバージョン番号が要求されたバージョン番号より低位である場合、アプリケーションはこの Windows LUA インプリメンテーションを使用することはできますが、要求されたバージョン番号でサポートされていない機能を使用することはできません。下位バージョンでは使用できない機能が必要なために使用せざるを得ない場合は、初期化が失敗し、LUA verb を発行することはできません。

szDescription

Windows LUA ソフトウェアを記述するテキスト・ストリング。

WinRUI

アプリケーションは、この関数を使用して RUI verb を発行します。verb が非同期で完了すると、LUA はアプリケーションのウィンドウ・ハンドルにメッセージを通知することによって完了を通知します。

最初に WinRUI 呼び出しを使用する前に、アプリケーションは、RegisterWindowMessage を使用して、LUA が非同期 verb 完了を示すメッセージに使用するメッセージ ID を取得しておく必要があります。詳しくは、23 ページの『使用法』を参照してください。

関数呼び出し

```
int WINAPI WinRUI (
    HWND hWnd,
    LUA_VERB_RECORD far * lpVCB
);
```

Windows アプリケーションの LUA エントリー・ポイント

LUA_VERB_RECORD 構造体の定義については、51 ページの『第 3 章 LUA VCB 構造体』を参照してください。

指定パラメーター

指定パラメーターは次のとおりです。

hWnd LUA が非同期 verb 完了を通知するメッセージの通知に使用するウィンドウ・ハンドルです。

lpVCB verb の VCB 構造体へのポインター。WinRUI 関数の場合、*lua_post_handle* パラメーターは予約済みなので、0 (ゼロ) のままにしておきます。

VCB 構造体についての詳細は、51 ページの『第 3 章 LUA VCB 構造体』を参照してください。個々の verb での使用方法についての詳細は、63 ページの『第 4 章 RUI verb』を参照してください。

注: LUA VCB には、「予約済み」としてマークされた多くのパラメーターが含まれています。これらの予約済みパラメーターには、CS/AIX ソフトウェアにより内部的に使用されるものがありますが、このバージョンでは使用されなくても将来のバージョンで使用されるものもあります。アプリケーションでは、これらの予約済みパラメーターには決してアクセスしないでください。verb によって使用される他のパラメーターをアプリケーションが設定する前に、VCB の内容全体をゼロに設定して、これらのパラメーターすべてを確実にゼロに設定しておく必要があります。このように設定しておくこと、CS/AIX がその内部使用パラメーターを誤って解釈することはありません。またこれにより、今後の CS/AIX のバージョンで、これらのパラメーターを使って新しい関数を引き続き使用できるようになります。

VCB の内容をゼロに設定するには、次のように `memset` を使用します。

```
memset(vcb, 0, sizeof(vcb));
```

戻り値

関数の戻り値は、以下のいずれかです。

0 (ゼロ)

関数呼び出しが受け入れられて、LUA verb が処理されます。アプリケーションは、23 ページの『同期および非同期の Verb の完了』で説明されているように、VCB 構造体内の *lua_flag2.async* パラメーターを検査して、verb がすでに同期して完了しているのか、あるいは非同期で完了するのかを判別します。

WLUAINVALIDHANDLE

指定された *hWnd* パラメーターが有効なウィンドウ・ハンドルではありませんでした。

WLUASTARTUPNOTCALLED

アプリケーションは WinRUIStartup 呼び出しを発行していませんが、これは、どの LUA verb を発行する前にも必要です。

VCB 構造体内に戻されるパラメーター情報については、63 ページの『第 4 章 RUI verb』に記載されている個々の verb の説明を参照してください。

使用法

最初に WinRUI を使用する前に、アプリケーションは、RegisterWindowMessage を使用して、LUA が非同期 verb 完了を示すメッセージに使用するメッセージ ID を取得しておく必要があります。RegisterWindowMessage は、標準の Windows 関数呼び出しであり、LUA に固有のものではありません。関数についての詳細は、Windows の文書を参照してください (後続の LUA verb を発行する前に、再度この呼び出しを発行する必要はありません。戻り値は、アプリケーションが発行するすべての呼び出しで同じです)。

アプリケーションは、関数にストリング「WinRUI」を渡さなければなりません。戻り値はメッセージ ID (RegisterWindowMessage 呼び出しから戻される値) です。

WinRUI エントリー・ポイントを使用して発行された LUA verb が非同期で完了するたびに、LUA は、WinRUI 呼び出しに指定されたウィンドウ・ハンドルにメッセージを通知します。メッセージの形式は、次のとおりです。

- メッセージ ID は、RegisterWindowMessage 呼び出しから戻された値です。
- *lParam* 引数には、元の WinRUI 呼び出しに指定された VCB のアドレスが含まれます。アプリケーションは、このアドレスを使用して、VCB 構造体に戻されたパラメーターにアクセスできます。
- *wParam* 引数は未定義です。

同期および非同期の Verb の完了

verb が発行されるとすぐに、LUA がその verb の処理をすべて完了できる場合があります。この場合、verb は同期復帰します。1 次戻りコードは LUA_IN_PROGRESS 以外の値に設定され、*lua_flag2.async* ビットは 0 (ゼロ) に設定されます (これらの戻りパラメーターについては、63 ページの『第 4 章 RUI verb』を参照してください)。

verb が非同期で戻れるようにするために、アプリケーションは、LUA エントリー・ポイントにウィンドウ・ハンドルを指定します。verb が同期して完了した場合、LUA はこのウィンドウ・ハンドルを使用しません。verb が非同期で完了した場合、LUA はこのウィンドウ・ハンドルにメッセージを通知することにより verb の完了を通知します。メッセージには元の VCB へのポインターが含まれます。

ある verb が同期して完了するのか、それとも非同期で完了するのかを、アプリケーション側で予測することはできません。

verb は、コールバックから発行することができますが、常に非同期で完了するとは限りません。このような verb がライブラリー内から発行されて、失敗した場合、同期で戻ることがあります。アプリケーションは、失敗した verb をコールバック内で再発行することはできません。

コールバック・コンテキストから並行して RUI_INIT を繰り返し発行すると、最終的には、RUI_INIT がメモリー・エラーで失敗します。ただし、アプリケーション・スレッドから verb を発行した場合は、すべてのシステム・メモリーが使用可能になり、より多くの発行が正常に完了します。

WinRUIGetLastInitStatus

アプリケーションは、この関数を使用して、直前の未解決の RUI_INIT verb の状況を判別します。戻された情報から、セッション開始を取り消すか (RUI_TERM の発行により)、あるいはセッションが確立されるのを待機するかを判断します。

この関数は、次のいずれかを行うために使用できます。

- 特定の RUI_INIT verb によって開始されたセッションの現在の状況に関する情報を要求する。
- 特定のセッションまたはすべてのセッションについて、セッション状況の変更の非同期通知を要求する。セッション状況が変更すると、LUA は、メッセージをアプリケーションのウィンドウ・ハンドルに通知するか、アプリケーションのイベント・ハンドルにシグナルを送ることによって、そのことを知らせます。
- セッション状況の変更の非同期通知に対する直前の要求を取り消す。

最初に WinRUI 呼び出しを使用する前に、アプリケーションは、WinRUIStartup を使用して Windows LUA アプリケーションとして登録する必要があります。状況の変更の非同期通知が必要である場合は、RegisterWindowMessage も使用して、LUA がその通知のために使用するメッセージ ID を取得する必要があります。これらの呼び出しについての詳細は、19 ページの『WinRUIStartup』および 26 ページの『使用法』を参照してください。

関数呼び出し

```
int WINAPI WinRUIGetLastInitStatus (  
    DWORD Sid,  
    HANDLE StatusHandle,  
    DWORD NotifyType,  
    BOOL ClearPrevious  
);
```

指定パラメーター

指定パラメーターは次のとおりです。

Sid 特定の RUI_INIT verb のセッション状況に関する情報を取得するか、その verb のセッション状況の変更に対する直前の通知要求を取り消すには、最初に RUI_INIT verb から戻ったときに返された セッション ID を指定します。

未解決のすべての RUI_INIT verb のセッション状況の変更についての通知を要求するには、0 (ゼロ) を指定します。この場合、情報が常に非同期で戻されるため、*StatusHandle* パラメーターは有効な Windows ハンドルを指定する必要があります。

未解決のすべての RUI_INIT verb のセッション状況の変更についての通知を取り消すには、0 (ゼロ) を指定します。

StatusHandle

以降の変更の通知は要求せずに、特定の RUI_INIT verb の現在のセッション状況を取得するには、ヌルのハンドルを指定します。

Windows アプリケーションの LUA エントリー・ポイント

セッション状況の変更に関する通知を要求するには、特定の RUI_INIT verb またはすべての未解決 RUI_INIT verb に対して、セッション状況が変更されたときに LUA が使用する Windows ハンドルまたはイベント・ハンドルを指定します。

直前の通知要求を取り消すために *ClearPrevious* パラメーターを TRUE に設定した場合、LUA は、このパラメーターを無視します。

NotifyType

非同期通知を要求している場合は、このパラメーターによって、非同期通知メッセージで LUA が RUI_INIT verb を識別する方法が決まります。指定できる値は次のとおりです。

WLUA_NTIFY_MSG_CORRELATOR

StatusHandle パラメーターには、ウィンドウ・ハンドルが含まれます。RUI_INIT verb に指定された *lua_correlator* の値によって verb を識別します。

WLUA_NTIFY_MSG_SID

StatusHandle パラメーターには、ウィンドウ・ハンドルが含まれます。RUI_INIT verb に戻された *lua_sid* の値によって verb を識別します。

WLUA_NTIFY_EVENT

StatusHandle パラメーターには、イベント・ハンドルが含まれます。

StatusHandle パラメーターがヌル (現在の状況情報を要求する) であるか、*ClearPrevious* パラメーターが TRUE に設定されている場合 (直前の通知要求を取り消す)、LUA はこのパラメーターを無視します。

ClearPrevious

直前の通知要求を取り消すには、このパラメーターを TRUE に設定します。LUA は、*StatusHandle* パラメーターと *ClearPrevious* パラメーターを無視します。現在の状況または将来の状況の変更通知を要求するには、このパラメーターを FALSE に設定します。

戻り値

関数が正常に完了した場合は、関数からの戻り値は次のいずれかです。

WLUALINKINACTIVE

ホストとの通信リンクはまだアクティブになっていません。

WLUAPUINACTIVE

ホストとの通信リンクはアクティブですが、物理装置活動化 (ACTPU) がまだ受信されていません。

WLUAPUACTIVE

ホストから ACTPU を受信しました。

WLUAPUREACTIVATED

ホストによって PU が再度アクティブにされました。

WLUALUINACTIVE

ホストとの通信リンクがアクティブで、物理装置活動化 (ACTPU) が受信されましたが、ACTLU がまだ受信されていません。

Windows アプリケーションの LUA エントリー・ポイント

WLUALUACTIVE

LU がアクティブです。

WLUALUREACTIVATED

LU が再度アクティブにされました。

WLUAGETLU

アプリケーションは、ノードとの接触を確立中です。

アプリケーションが状況変更の通知を要求すると、状況が変更するたびに、アプリケーションに送信される Windows メッセージにこれらの値のいずれか 1 つが含まれます。詳しくは、『使用法』を参照してください。

以下の戻り値は、正常に実行されなかった関数を示しています。

WLUASYSNOTREADY

SNA ソフトウェアが実行していません。

WLUANTFYINVALID

NotifyType パラメーターが無効な値に設定されました。

WLUAINVALIDHANDLE

指定された *StatusHandle* パラメーターが有効なウィンドウ・ハンドルではありませんでした。

WLUASTARTUPNOTCALLED

アプリケーションは WinRUIStartup 呼び出しを発行していませんが、これは、どの LUA verb を発行する前にも必要です。

WLUAUNKNOWN

内部エラー。セッション状況が不明です。

WLUASIDINVALID

指定された *Sid* パラメーターが未解決の RUI_INIT verb のセッション ID と一致しませんでした。

WLUASIDZERO

アプリケーションがゼロのセッション ID (すべてのセッションを示す) を指定しましたが、Windows ハンドル (非同期通知を示す) も TRUE という *ClearPrevious* 値 (直前の通知要求をクリアする) も指定されませんでした。

WLUAGLOBALHANDLER

アプリケーションは、すべての RUI_INIT verb に対する状況変更の通知を以前に要求していますが、最初に「すべてのセッション」の通知をクリアしない限り、特定セッションに対する通知を要求することはできません。

使用法

アプリケーションが、Windows メッセージを使用して、状況変更の非同期通知を要求している場合、状況変更を知らせるメッセージに使用するメッセージ ID を LUA が取得するためには、初めて WinRUIGetLastInitStatus 呼び出しを使用する前に、RegisterWindowMessage 呼び出しを使用する必要があります。

RegisterWindowMessage 呼び出しは、標準の Windows 関数呼び出しであり、LUA に固有のものではありません。関数についての詳細は、Windows の文書を参照して

ください (この関数に対する後続の呼び出しを発行する前に、再度この呼び出しを発行する必要はありません。戻り値は、アプリケーションが発行するすべての呼び出しと同じです)。

アプリケーションは、関数にストリング「WinRUI」を渡さなければなりません。戻り値はメッセージ ID (RegisterWindowMessage 呼び出しから戻される値) です。

セッション状況が変更されるたびに、LUA は、WinRUI 呼び出しに指定されたウィンドウ・ハンドルにメッセージを通知します。メッセージの形式は、次のとおりです。

- メッセージ ID は、RegisterWindowMessage 呼び出しから戻された値です。
- *lParam* 引数には、*NotifyType* パラメーターの定義に従って、元の RUI_INIT verb に指定された相関係数の値、または元の RUI_INIT verb に戻されるセッション ID が含まれます。アプリケーションは、この値を使用して、メッセージと元の verb を相互に関連付けることができます。
- *wParam* 引数には、セッション状況 (25 ページの『戻り値』で、正常な実行に対してリストされている値の 1 つ)、または処理中に内部エラーが起きた場合は値 WLUAUNKNOWN が含まれます。

アプリケーションが、イベント・ハンドルを使用して状況変更の非同期通知を要求している場合、インプリメントは次のようになります。

```
WinRUIGetLastInitStatus(Sid,EventHandle,WLUA_NOTIFY_EVENT,FALSE);
```

ハンドルが提供されたイベントは、状況の変更が起きるとシグナル状態になります。イベントがシグナル状態であると情報が戻されないため、状況を判別するには、次のように、さらに呼び出しを発行する必要があります。

```
Status = WinRUIGetLastInitStatus(Sid,NULL,0,FALSE);
```

注: この例では、*Sid* を指定する必要があります。

WinRUICleanup

アプリケーションは、RUI verb の発行を終了してから、この関数を使用して、Windows RUI ユーザーとしての登録を抹消します。

関数呼び出し

```
BOOL WINAPI WinRUICleanup (void);
```

指定パラメーター

この関数には指定パラメーターはありません。

戻り値

関数の戻り値は、以下のいずれかです。

TRUE アプリケーションは正常に登録抹消されました。

FALSE 呼び出しの処理中にエラーが発生しました。アプリケーションは登録抹消されませんでした。ログ・ファイルを確認して、エラーの原因を示すメッセージを参照してください。

GetLuaReturnCode

アプリケーションは、この関数を使用して、指定した VCB からの 1 次および 2 次戻りコードを示す印刷可能文字ストリングを取得します。この文字ストリングを使用して、LUA_OK 以外の戻りコードに関するアプリケーション・エラー・メッセージを生成できます。

関数呼び出し

```
int WINAPI GetLuaReturnCode (
    struct LUA_COMMON FAR * vcbptr,
    unsigned int           buffer_length,
    unsigned char far *    buffer_addr
);
```

指定パラメーター

指定パラメーターは次のとおりです。

vcbptr verb の VCB 構造体へのポインター。VCB 構造体および個々の verb での使用法についての詳細は、63 ページの『第 4 章 RUI verb』を参照してください。

buffer_length

戻されたデータ・ストリングを保持するためにアプリケーションが提供するバッファの長さ (バイト)。推奨の長さは 256 バイトです。

buffer_addr

戻されたデータ・ストリングを保持するためにアプリケーションが提供するバッファのアドレス。

戻り値

関数の戻り値は、以下のいずれかです。

0 (ゼロ)

関数の処理が正常に完了しました。

0x20000001

LUA は、指定された VCB からの読み取り、または指定されたデータ・バッファへの書き込みができませんでした。

0x20000002

指定されたデータ・バッファは、戻された文字ストリングを保持するのに小さすぎます。

0x20000003

この関数に対する戻り文字ストリングを生成するダイナミック・リンク・ライブラリー **LUASTR32.DLL** をロードできませんでした。

戻り値が 0 (ゼロ) である場合、戻される文字ストリングは *buffer_addr* パラメーターで示されるバッファに入っています。このストリングの最後はヌル文字 (2 進ゼロ) ですが、末尾の改行 ($\backslash n$) 文字は含みません。

SLI

アプリケーションは、この関数を使用して、LUA SLI verb を発行します。verb が非同期で完了すると、LUA は、アプリケーションが提供するイベント・ハンドルにシグナルを送ることによって完了を通知します。

関数呼び出し

```
void WINAPI SLI(verb)
LUA_VERB_RECORD FAR * verb;
```

指定パラメーター

指定パラメーターは次のとおりです。

verb 発行される verb のパラメーターが入っている verb 制御ブロック (VCB) を指すポインター。VCB 構造体は、LUA ヘッダー・ファイル **winlua.h** で定義されます。このファイルは、Windows クライアント・ソフトウェアのインストール先ディレクトリー内の、サブディレクトリー **/sdk** にインストールされます。VCB 構造体についての詳細は、51 ページの『第 3 章 LUA VCB 構造体』を参照してください。

注: LUA VCB には、「予約済み」としてマークされた多くのパラメーターが含まれています。これらの予約済みパラメーターには、CS/AIX ソフトウェアにより内部的に使用されるものがありますが、このバージョンでは使用されなくても将来のバージョンで使用されるものもあります。アプリケーションでは、これらの予約済みパラメーターには決してアクセスしないでください。verb によって使用される他のパラメーターをアプリケーションが設定する前に、VCB の内容全体をゼロに設定して、これらのパラメーターすべてを確実にゼロに設定しておく必要があります。このように設定しておくこと、CS/AIX がその内部使用パラメーターを誤って解釈することはありません。またこれにより、今後の CS/AIX のバージョンで、これらのパラメーターを使って新しい関数を引き続き使用できるようになります。

VCB の内容をゼロに設定するには、次のように `memset` を使用します。

```
memset(vcb, 0, sizeof(vcb));
```

戻り値

このエントリー・ポイントは値を戻しません。呼び出しから制御が戻されると、アプリケーションは VCB 内のパラメーターを調べて、その verb が同期して完了したのか、非同期で完了するのかを確認できます。詳しくは、『使用法』を参照してください。

使用法

verb が発行されるとすぐに、LUA がその verb の処理をすべて完了できる場合があります。この場合、verb は同期復帰します。1 次戻りコードは `LUA_IN_PROGRESS` 以外の値に設定され、`lua_flag2.async` ビットは 0 (ゼロ) に設定されます (これらの戻りパラメーターについては、115 ページの『第 5 章 SLI verb』を参照してください)。

Windows アプリケーションの LUA エントリー・ポイント

また一方では、LUA は `verb` を完了する前に、リモート LU またはノードからの情報を待つ必要がある場合もあります。この場合、`verb` は非同期復帰します。1 次戻りコードは `LUA_IN_PROGRESS` に設定され、`lua_flag2.async` ビットは 1 に設定されます。アプリケーションは、この時点で別の処理を実行することもできますし、`verb` が完了したという LUA からの通知を待つこともできます。LUA は、1 次戻りコードをその最終値に設定し、`lua_flag2.async` ビットを 1 に設定したままで、この通知を発行します。

アプリケーションは、提供する VCB の一部として、`lua_post_handle` パラメーターでイベント・ハンドルを提供します。イベントは非シグナル状態で、ハンドルは、そのイベントに対する `EVENT_MODIFY_STATE` アクセスが可能でなければなりません。`verb` が同期して完了した場合は、LUA はそのイベント・ハンドルにシグナルを送りません。`verb` が非同期で完了すると、LUA は、イベント・ハンドルにシグナルを送ることによって完了を通知します。

アプリケーションは、イベント・ハンドルを待機するために `WaitForSingleObject` または `WaitForMultipleObject` 呼び出しを発行します。イベントがシグナル状態になると、アプリケーションは、1 次戻りコードと 2 次戻りコードのエラー検査を行います。

ある `verb` が同期して完了するのか、それとも非同期で完了するのかを、アプリケーション側で予測することはできません。

WinSLIStartup

アプリケーションはこの関数を使用して、Windows SLI ユーザーとして登録し、LUA ソフトウェアがアプリケーションに必要な Windows LUA バージョンをサポートしているかどうかを判別します。

関数呼び出し

```
int WINAPI WinSLIStartup (
    WORD wVersionRequired;
    LUADATA far * lpData;
)

typedef struct
{
    WORD wVersion;
    char szDescription[41];
} LUADATA;
```

指定パラメーター

指定パラメーターは次のとおりです。

wVersionRequired

アプリケーションに必要な Windows LUA のバージョン。CS/AIX は、バージョン 1.0 をサポートしています。

下位バイトには、メジャー・バージョン番号を指定し、高位バイトにはマイナー・バージョン番号を指定します。たとえば、次のように指定します。

バージョン	wVersionRequired
1.0	0x0001
1.1	0x0101
2.0	0x0002

アプリケーションが複数のバージョンを使用する場合、使用できる最高位のバージョンを指定する必要があります。

戻り値

関数の戻り値は、以下のいずれかです。

0 (ゼロ)

アプリケーションは正常に登録されており、Windows LUA ソフトウェアは、アプリケーションによって指定されたバージョン番号、あるいはその下位バージョンのどちらかをサポートしています。アプリケーションは、LUADATA 構造体内のバージョン番号を検査して、それが十分なレベルであることを確認する必要があります。

WLUAVERNOTSUPPORTED

アプリケーションで指定されたバージョン番号が、Windows LUA ソフトウェアでサポートされていません。アプリケーションは登録されませんでした。

WLUAINITREJECT

アプリケーションはすでに WinSLIStartup を呼び出しており、正常に登録されています。この関数を複数回呼び出すことはできません。

WLUASYSNOTREADY

CS/AIX ソフトウェアが開始されなかったか、ローカル・ノードがアクティブではありません。アプリケーションは登録されませんでした。

WLUAFailure

Windows LUA ソフトウェアの初期化中に、オペレーティング・システムでエラーが発生しました。アプリケーションは登録されませんでした。ログ・ファイルを確認して、エラーの原因を示すメッセージを参照してください。

WinSLIStartup からの戻り値が 0 (ゼロ) である場合、LUADATA 構造体に、Windows LUA ソフトウェアによって提供されるサポート情報が含まれています。戻り値がゼロ以外の値である場合、この構造体の内容は未定義で、アプリケーションは構造体の内容をチェックする必要はありません。この構造体にあるパラメーターは次のとおりです。

wVersion

ソフトウェアがサポートする Windows LUA バージョン番号で、*wVersionRequired* パラメーターと同じ形式です (20 ページの『指定パラメーター』を参照してください)。CS/AIX は、バージョン 1.0 をサポートしています。

ソフトウェアが要求されるバージョン番号をサポートする場合、このパラメーターは *wVersionRequired* パラメーターと同一の値に設定されます。そうでない場合、ソフトウェアがサポートする最高位のバージョンに設定されますが、これは、アプリケーションに指定されるバージョン番号より低位になります。アプリケーションは戻り値を検査し、以下のようなアクションを行います。

Windows アプリケーションの LUA エントリー・ポイント

- 戻されたバージョン番号が要求されたバージョン番号と同一である場合、アプリケーションはこの Windows LUA インプリメンテーションを使用することができます。
- 戻されたバージョン番号が要求されたバージョン番号より低位である場合、アプリケーションはこの Windows LUA インプリメンテーションを使用しますが、要求されたバージョン番号にサポートされない機能を使用することはできません。下位バージョンでは使用できない機能が必要なために使用せざるを得ない場合は、初期化が失敗し、LUA verb を発行することはできません。

szDescription

Windows LUA ソフトウェアを記述するテキスト・ストリング。

WinSLI

アプリケーションは、この関数を使用して、SLI verb を発行します。verb が非同期で完了すると、LUA はアプリケーションのウィンドウ・ハンドルにメッセージを通知することによって完了を通知します。

最初に WinSLI 呼び出しを使用する前に、アプリケーションは、RegisterWindowMessage を使用して、LUA が非同期 verb 完了を示すメッセージに使用するメッセージ ID を取得しておく必要があります。詳しくは、33 ページの『使用法』を参照してください。

関数呼び出し

```
int WINAPI WinSLI (
    HWND hWnd,
    LUA_VERB_RECORD far * lpVCB
);
```

LUA_VERB_RECORD 構造体の定義については、51 ページの『第 3 章 LUA VCB 構造体』を参照してください。

指定パラメーター

指定パラメーターは次のとおりです。

hWnd LUA が非同期 verb 完了を通知するメッセージの通知に使用するウィンドウ・ハンドルです。

lpVCB verb の VCB 構造体へのポインター。WinSLI 関数の場合、*lua_post_handle* パラメーターは予約済みなので、0 (ゼロ) のままにしておきます。

VCB 構造体についての詳細は、51 ページの『第 3 章 LUA VCB 構造体』を参照してください。個々の verb での使用法についての詳細は、115 ページの『第 5 章 SLI verb』を参照してください。

注: LUA VCB には、「予約済み」としてマークされた多くのパラメーターが含まれています。これらの予約済みパラメーターには、CS/AIX ソフトウェアにより内部的に使用されるものがありますが、このバージョンでは使用されなくても将来のバージョンで使用されるものもあります。アプリケーションでは、これらの予約済みパラメーターには決してアクセスしないでください。verb によって使用される他のパラメーターをアプリケーションが設定する前に、VCB の内容全体をゼロに設定して、

Windows アプリケーションの LUA エントリー・ポイント

これらのパラメーターすべてを確実にゼロに設定しておく必要があります。このように設定しておくこと、CS/AIX がその内部使用パラメーターを誤って解釈することはありません。またこれにより、今後の CS/AIX のバージョンで、これらのパラメーターを使って新しい関数を引き続き使用できるようになります。

VCB の内容をゼロに設定するには、次のように `memset` を使用します。

```
memset(vcb, 0, sizeof(vcb));
```

戻り値

関数の戻り値は、以下のいずれかです。

0 (ゼロ)

関数呼び出しが受け入れられて、LUA verb が処理されます。アプリケーションは、34 ページの『同期および非同期の Verb の完了』で説明されているように、VCB 構造体内の `lua_flag2.async` パラメーターを検査して、verb がすでに同期して完了しているのか、あるいは非同期で完了するのかを判別します。

WLUAINVALIDHANDLE

指定された `hWnd` パラメーターが有効なウィンドウ・ハンドルではありませんでした。

WLUASTARTUPNOTCALLED

アプリケーションは `WinSLIStartup` 呼び出しを発行していませんが、これは、どの SLI verb を発行する前にも必要です。

VCB 構造体内に戻されるパラメーター情報については、115 ページの『第 5 章 SLI verb』に記載されている個々の verb の説明を参照してください。

使用法

最初に `WinSLI` を使用する前に、アプリケーションは、`RegisterWindowMessage` 呼び出しを使用して、LUA が非同期 verb 完了を示すメッセージに使用するメッセージ ID を取得しておく必要があります。`RegisterWindowMessage` は、標準の Windows 関数呼び出しであり、LUA に固有のものではありません。関数についての詳細は、Windows の文書を参照してください (後続の LUA verb を発行する前に、再度この呼び出しを発行する必要はありません。戻り値は、アプリケーションが発行するすべての呼び出しと同じです)。

アプリケーションは、関数にストリング `WinSLI` を渡さなければなりません。戻り値はメッセージ ID (`RegisterWindowMessage` 呼び出しから戻される値) です。

`WinSLI` エントリー・ポイントを使用して発行された LUA verb が非同期的に完了するたびに、LUA は、`WinSLI` 呼び出しに指定されたウィンドウ・ハンドルにメッセージをポストします。メッセージの形式は、次のとおりです。

- メッセージ ID は、`RegisterWindowMessage` 呼び出しから戻された値です。
- `lParam` 引数には、元の `WinSLI` 呼び出しに指定された VCB のアドレスが含まれます。アプリケーションは、このアドレスを使用して、VCB 構造体に戻されたパラメーターにアクセスできます。

Windows アプリケーションの LUA エントリー・ポイント

- *wParam* 引数は未定義です。

同期および非同期の Verb の完了

verb が発行されるとすぐに、LUA がその *verb* の処理をすべて完了できる場合があります。この場合、*verb* は同期復帰します。1 次戻りコードは `LUA_IN_PROGRESS` 以外の値に設定され、*lua_flag2.async* ビットは 0 (ゼロ) に設定されます (これらの戻りパラメーターについては、115 ページの『第 5 章 SLI verb』を参照してください)。

verb が非同期で戻れるようにするために、アプリケーションは、LUA エントリー・ポイントにウィンドウ・ハンドルを指定します。 *verb* が同期して完了した場合、LUA はこのウィンドウ・ハンドルを使用しません。 *verb* が非同期で完了した場合、LUA はこのウィンドウ・ハンドルにメッセージを通知することにより *verb* の完了を通知します。メッセージには元の VCB へのポインターが含まれます。

ある *verb* が同期して完了するのか、それとも非同期で完了するのかを、アプリケーション側で予測することはできません。

verb は、コールバックから発行することができますが、常に非同期で完了するとは限りません。このような *verb* がライブラリー内から発行されて、失敗した場合、同期で戻ることがあります。アプリケーションは、失敗した *verb* をコールバック内で再発行することはできません。

コールバック・コンテキストから並行して繰り返し `SLI_OPEN` を発行すると、最終的には、`SLI_OPEN` がメモリー・エラーで失敗します。ただし、アプリケーション・スレッドから *verb* を発行した場合は、すべてのシステム・メモリーが使用可能になり、より多くの発行が正常に完了します。

WinSLICleanup

アプリケーションは、SLI verb の発行が終了してから、この関数を使用して、Windows SLI ユーザーとしての登録を抹消します。

関数呼び出し

```
BOOL WINAPI WinSLICleanup (void);
```

指定パラメーター

この関数には指定パラメーターはありません。

戻り値

関数の戻り値は、以下のいずれかです。

TRUE アプリケーションは正常に登録抹消されました。

FALSE 呼び出しの処理中にエラーが発生しました。アプリケーションは登録抹消されませんでした。ログ・ファイルを確認して、エラーの原因を示すメッセージを参照してください。



LUA verb の発行

LUA verb の発行に必要なステップは、次のとおりです。以下の例は、RUI_INIT verb の使用法を示しています。

1. アプリケーションのソース・コードに LUA ヘッダー・ファイルを組み込みます。

AIX, LINUX

```
#include < lua_c.h>
```

WINDOWS

```
#include < winlua.h >
```



AIX, LINUX

2. verb が非同期で完了したことを示すために LUA が使用するコールバック関数を設定します (詳しくは、13 ページの『AIX または Linux アプリケーションの LUA エントリー・ポイント』を参照してください)。

```
void callback(verb)
LUA_VERB_RECORD * verb;
{
    .
    .
    .
}
```

WINDOWS

これがアプリケーションから発行された最初の verb であり、アプリケーションが WinRUI 呼び出しを使用して RUI verb を発行する予定であれば、WinRUIStartup 呼び出しを発行して、アプリケーションが LUA を使用するための初期化を行ってください。同様に、アプリケーションが WinSLI 呼び出しを使用して SLI verb を発行するのであれば、WinSLIStartup 呼び出しを発行して、アプリケーションが LUA を使用するための初期化を行ってください (詳しくは、16 ページの『Windows アプリケーションの LUA エントリー・ポイント』を参照してください)。この呼び出しは、アプリケーションの最初の LUA verb の前に 1 回、発行する必要がある、後続の verb の前に繰り返し発行することはできません。

RegisterWindowMessage 呼び出しも発行し、LUA verb の完了を知らせるメッセージを通知するときに LUA が使用するメッセージ ID を取得します (詳しくは、16 ページの『Windows アプリケーションの LUA エントリー・ポイント』

を参照してください)。この呼び出しは、アプリケーションの最初の LUA verb の前に 1 回、発行する必要があり、後続の verb の前に繰り返し発行する必要はありません。



- VCB 構造体の変数を作成します。

```
LUA_VERB_RECORD rui_init;
```

LUA_VERB_RECORD 構造体は、ヘッダー・ファイル **lua_c.h** (AIX または Linux アプリケーション) または **winlua.h** (Windows アプリケーション) で宣言されています。VCB 構造体については、51 ページの『第 3 章 LUA VCB 構造体』を参照してください。

- VCB 内の変数をクリアします (0 に設定します)。

```
memset( rui_init, 0, sizeof( rui_init) );
```

LUA では、すべての予約パラメーターと、現在発行しようとしている verb が必要としないすべてのパラメーターを 0 (ゼロ) に設定する必要があります。予約パラメーターの詳細については、51 ページの『LUA verb 制御ブロック (VCB) の形式』を参照してください。最も簡単な方法は、VCB 全体をゼロに設定してから、この verb に必要なパラメーターを設定する方法です。

- LUA に情報を提供する VCB パラメーターに値を割り当てます。

```
rui_init.common.lua_verb = LUA_VERB_RUI  
rui_init.common.lua_verb_length = sizeof(LUA_COMMON);  
rui_init.common.lua_opcode = LUA_OPCODE_RUI_INIT;  
memcpy (rui_init.common.lua_luname, "THISLU ", 8);
```

AIX、LINUX

```
rui_init.common.lua_post_handle = (unsigned long) callback;
```

WINDOWS

rui_init.common.lua_post_handle パラメーターは予約済みなので、0 (ゼロ) のままにしておきます。



値 **LUA_VERB_RUI** と **LUA_OPCODE_RUI_INIT** はシンボリック定数です。これらの定数は、ヘッダー・ファイル **lua_c.h** (AIX または Linux アプリケーション) または **winlua.h** (Windows アプリケーション) で定義されています。異種システム間の移植性を保つため、整数値ではなく、シンボリック定数の使用をお勧めします (詳しくは、48 ページの『移植可能なアプリケーションの作成』を参照してください)。

- LUA を呼び出します。この関数呼び出しのパラメーターには、VCB 構造体のアドレスを使用します。

AIX, LINUX

```
RUI ( &rui_init );
```

WINDOWS

WinRUI エントリー・ポイントには、さらにパラメーターが必要です。これは、LUA が、verb の非同期完了を示すメッセージを通知するウィンドウのウィンドウ・ハンドルです。

```
WinRUI ( handle, (LUA_VERB_RECORD far *) &rui_init );
```

7. *lua_flag2.async* パラメーターを調べて、verb が同期して完了したか、あるいは非同期で完了したかを明確にします。

```
if (rui_init.common.lua_flag2.async )
{
    /* verb will complete asynchronously */
    /* using the supplied callback routine */
    /* continue with other processing */
    .
    .
}
else
{
    /* verb has completed synchronously */
    /* callback routine will not be called */
    /* process the returned values here */
    .
    .
}
```

lua_flag2.async パラメーターが、verb が非同期で完了することを示している場合、その時点で VCB 内にある他のパラメーターにプログラムの実行のメイン・スレッドがアクセスすることはできません。LUA がコールバック・ルーチンを呼び出すと、アプリケーションは VCB パラメーターにアクセスできるようになります。

8. LUA から戻された変数を使用します。7 のステップで verb が非同期完了することが確認された場合は、verb が完了するまでこのステップは実行しないでください。AIX または Linux システムでは、この処理は通常、コールバック・ルーチンによって行われます。7 のステップで verb の同期完了が確認された場合、コールバック・ルーチンは呼び出されないの、メイン・コード・パスでこの処理を行ってください。

```
if( rui_init.common.lua_prim_rc == LUA_OK )
{
    /* Init OK */
    .
    .
}
```

```
else
{
    /* Do error routine */
    .
    .
}
}
```

SNA 情報

この節では、ホストとの通信用に CS/AIX LUA アプリケーションを作成する際に考慮しなければならない SNA 情報について説明します。ダウンストリーム LU との通信用に 1 次 RUI アプリケーションを作成する場合は、42 ページの『1 次 RUI の SNA 情報』を参照してください。

本書には、SNA の概念の詳しい説明は記載していません。SNA のメッセージ・フローの詳細については、設計する CS/AIX LUA アプリケーションのホスト・アプリケーションの資料を参照してください。

BIND 検査: RUI

ホストは、LU セッションの初期化時に、その LU セッションで使用する RU サイズなどの情報を示す BIND メッセージを CS/AIX LUA アプリケーションに送信します。CS/AIX は、RUI_READ verb で、このメッセージを LUA アプリケーションに戻します。BIND で指定されたパラメーターが適切かどうかは、LUA アプリケーション側で調べます。アプリケーションでの処理は、次の中から選択できます。

- BIND をそのまま受け入れる。この場合は、BIND に対する OK 応答を設定した RUI_WRITE verb を発行します。この応答では、データを送信する必要はありません。
- 1 つ以上の BIND パラメーターについて折衝する (これが可能なのは、BIND が折衝可能な場合だけです)。この場合、アプリケーションは、OK 応答を設定した RUI_WRITE verb を発行するだけでなく、変更された BIND をデータとして組み込みます。
- BIND をリジェクトする。この場合は、適切な SNA センス・コードをデータとして使用して、否定応答を含んだ RUI_WRITE verb を発行します。

RUI_WRITE verb の詳細については、63 ページの『第 4 章 RUI verb』を参照してください。

BIND パラメーターの妥当性検査と、送信されるすべてのメッセージと BIND パラメーターとの整合性の確保は、LUA アプリケーションが行います。ただし、次の 2 つの制約事項があります。

- CS/AIX は、BIND で指定されたサイズより大きい RU 長を指定した RUI_WRITE verb をすべてリジェクトします。
- CS/AIX では、2 次 LU がコンテンション勝者であり、エラー・リカバリーはコンテンション敗者が行うことを、BIND で指定する必要があります。

BIND 検査: SLI

ホストは、LU セッションの初期化時に、その LU セッションで使用する RU サイズなどの情報を示す BIND メッセージを CS/AIX LUA アプリケーションに送信します。

アプリケーションでは、ホストからの BIND 要求を処理するための独自のルーチンのアドレスをオプションで SLI_OPEN verb に指定できます。アプリケーションでこのように指定されていると、LUA は次のように、アプリケーションが提供するルーチンに追加の verb SLI_BIND_ROUTINE を送信して、そのルーチンが要求を処理できるようにします。BIND で指定されたパラメーターが適切かどうかは、LUA アプリケーション側で調べます。アプリケーションでの処理は、次の中から選択できます。

- BIND をそのまま受け入れる。そのためには、1 次戻りコード OK を設定した SLI_BIND_ROUTINE verb を戻します。アプリケーションでは、BIND を含むデータ・バッファは変更しません。
- 1 つ以上の BIND パラメーターについて折衝する (これが可能なのは、BIND が折衝可能な場合だけです)。そのためには、アプリケーションは、1 次戻りコード OK を設定し、データ・バッファには変更済み BIND を入れて、SLI_BIND_ROUTINE verb を戻します。
- BIND をリジェクトする。そのためには、1 次戻りコードとして LUA_NEGATIVE_RESPONSE を設定して SLI_BIND_ROUTINE verb を戻し、データ・バッファ内の BIND 要求を適切な SNA センス・コードで置き換えます。

BIND パラメーターの妥当性検査と、送信されるすべてのメッセージと BIND パラメーターとの整合性の確保は、LUA アプリケーションが行います。ただし、CS/AIX では、2 次 LU がコンテンション勝者であり、エラー回復はコンテンション敗者が行うことを、BIND で指定する必要があります。

否定応答と SNA センス・コード

SNA センス・コードが LUA アプリケーションに戻されるのは、次のような場合です。

- ホストが、LUA アプリケーションからの要求に対して否定応答を送信する場合。これには、その否定応答の理由を示す SNA センス・コードが含まれています。これは、次のように後続の RUI_READ verb または SLI_RECEIVE verb でアプリケーションに報告されます。
 - 1 次戻りコードを LUA_OK にする。
 - 要求応答インディケータ、応答タイプ・インディケータ、およびセンス・データ・インクルード・インディケータを、すべて 1 に設定する。これは、センス・データを含む否定応答であることを示します。
 - RUI_READ verb または SLI_RECEIVE verb で戻されるデータは SNA センス・コードである。
- ホストから無効なデータを受け取った場合、CS/AIX は通常、ホストに否定応答を送り、LUA アプリケーションにはその無効なデータを渡しません。これは、後続の RUI_READ verb または RUI_BID verb あるいは SLI_RECEIVE/SLI_BID で、アプリケーションに次のように報告されます。
 - 1 次戻りコードを LUA_NEGATIVE_RSP にする。

- 2 次戻りコードを、ホストに送られた SNA センス・コードにする。
- ホストから提供されたデータが無効であることを CS/AIX が検出しても、送信すべき正しいセンス・コードを決定できない場合があります。この場合、次のように、RUI_READ verb または SLI_RECEIVE verb で、無効なデータを例外要求 (EXR) の形で LUA アプリケーションに渡します。
 - 要求応答インディケータを 0 (ゼロ) に設定する。これは、要求であることを示します。
 - センス・データ・インクルード・インディケータを 1 に設定する。これは、センス・データが含まれていることを示します (このインディケータは通常、要求の場合にのみ使用されます)。
 - メッセージ・データが、提案する SNA センス・コードで置き換える。

アプリケーションは次に、このメッセージに対する否定応答を送信する必要があります。その場合、CS/AIX によって提案されたセンス・コードを使用することも、そのセンス・コードを変更することもできます。

- CS/AIX は、アプリケーションにセンス・コードを送信して、アプリケーションによって指定されたデータが無効であることを示す場合があります。これは、データを提供したアプリケーションに対して RUI_WRITE verb または SLI_SEND verb で以下のように報告されます。
 - 1 次戻りコードが LUA_UNSUCCESSFUL である。
 - 2 次戻りコードが、SNA センス・コードである。

SNA センス・コードと他の 2 次戻りコードとの区別

注: LUA 2 次戻りコードで使用されるバイト配列は、数値の最上位バイトは最後のバイトであり、最初のバイトではないことを意味します。

センス・コード以外の 2 次戻りコードの場合、上位 2 バイトは常に 0 (ゼロ) です。たとえば、0x01000000 (LUA_INVALID_LUNAME) は標準の LUA 2 次戻りコードであり、センス・コードではありません。

SNA センス・コードの場合、上位 2 バイトはゼロ以外です。つまり、最上位バイトでそのセンス・コードのカテゴリを示し、次のバイトでカテゴリ内のどのセンス・コードかを特定します (残りのバイトと 4 つ目のバイトには追加情報を入れるか、または 0 にすることができます)。たとえば、0x00000108 (LUA_RESOURCE_NOT_AVAILABLE) はセンス・コードです。

SNA センス・コードであるものも含め、すべての LUA 2 次戻りコードを 177 ページの『付録 A. 戻りコードの値』に示してあります。

SNA センス・コードに関する情報

戻されたセンス・コードに関する情報が必要な場合は、IBM の「*Systems Network Architecture: Formats*」を参照してください。この資料には、カテゴリごとに、センス・コードの番号順のリストが掲載されています。

また、CS/AIX コンピューター上で生成された特定の SNA センス・コードに関するオンライン・ヘルプ情報を取得することもできます。それには、コマンド行で **sna**

`-getsense` と入力し、続けてカテゴリと修飾子 (先頭の 4 桁) またはセンス・コード全体 (全 8 桁) を入力します。詳しくは、「*Communications Server for AIX Diagnostics Guide*」を参照してください。

ペーシング

ペーシングは LUA インターフェースによって処理されます。LUA アプリケーションはペーシングを制御する必要はないので、ペーシング・インディケータ・フラグは設定しないでください。

LUA アプリケーションからホストに送られたデータに対してペーシングが使用される場合 (これは BIND で決定されます)、RUI_WRITE verb または SLI_SEND verb の完了にしばらく時間がかかることがあります。これは、CS/AIX がさらにデータを送信するには、ホストからのペーシング応答を待たなければならないからです。

LUA アプリケーションを使用して 1 方向 (ホストへ、またはホストから) に大量のデータを転送する場合 (ファイル転送アプリケーションなど)、ホスト設定でその方向でのペーシングの使用を指定する必要があります。これは、データを受信するノードがデータであふれたり、データ・ストレージが不足したりしないようにするためです。

セグメント化

RU のセグメント化は、LUA インターフェースによって処理されます。LUA は、常にすべての RU をアプリケーションに渡します。アプリケーションは、すべての RU を LUA に渡す必要があります。

非標準ホストの応答/要求ヘッダー (RH) ビットの変更

BB (ブラケット開始) と RQE (例外要求) オプションを指定し、EB (ブラケット終了) オプションを指定せずに (ブラケットを開始して例外応答を発行するが、ブラケットを終了しないで)、ホストから LUA アプリケーションにデータが送信されることがあります。このようなオプションの組み合わせは、SNA では厳密にいうと無効ですが、一部のホスト・アプリケーションでは使用されます。

このようなホスト・アプリケーションをサポートするために、CS/AIX は、例外応答ではなく確定応答を指定するようにそのホスト・データを変更してから、アプリケーションに送信します。

肯定応答

CS/AIX は、アプリケーションから送信された応答を適切な要求と関連させるために、ホストから受け取った要求の記録を取ります。アプリケーションが応答を送信すると、CS/AIX は、その応答を元の要求のデータと関連させます。その後、その要求に関連付けられていたストレージを解放することができます。

ホストで例外応答のみ (否定応答は送信できるが、肯定応答は送信しない) が指定された場合でも、CS/AIX は、アプリケーションが後で否定応答を送信する場合に備えて、要求の記録を取ります。アプリケーションが応答を送信しないと、その要求に関連付けられているストレージは解放できません。

このため、CS/AIX では、ホストからの例外応答のみという要求に対して、LUA アプリケーションが肯定応答を発行することができるようになっていきます (これを肯定応答と呼びます)。この応答はホストには送信されず、要求に関連付けられたストレージをクリアするために CS/AIX によって使用されます。

チェーン終了までのデータの除去

ホストが要求単位 (RU) のチェーンを LUA アプリケーションに送信する場合、アプリケーションは、チェーン内の最後の RU を受信してから応答を送信することも、チェーンの最後の RU 以外の RU に対して否定応答を送信することもできます。チェーンの途中で否定応答が送信されると、CS/AIX は後続のすべての RU をそのチェーンから除去し、アプリケーションには送信しません。

CS/AIX は、チェーン内の最後の RU を受信すると、そのことをアプリケーションに知らせるために、RUI_READ verb または RUI_BID verb、あるいは SLI_RECEIVE/SLI_BID の 1 次戻りコードを LUA_NEGATIVE_RSP に設定し、2 次戻りコードを 0 (ゼロ) に設定します。

ホストは、チェーンの途中で CANCEL などのメッセージを送信することによって、チェーンを終了する場合があります。この場合、RUI_READ verb または SLI_RECEIVE verb で CANCEL がアプリケーションに戻され、LUA_NEGATIVE_RSP 戻りコード (39 ページの『否定応答と SNA センス・コード』を参照) は使用されません。

1 次 RUI の SNA 情報

この節では、ダウンストリーム LU との通信用に CS/AIX RUI 1 次側アプリケーションを作成する際に考慮しなければならない SNA 情報について説明します。

本書には、SNA の概念の詳しい説明は記載していません。SNA のメッセージ・フローの詳細については、設計する CS/AIX LUA アプリケーションのホスト・アプリケーションの資料を参照してください。

RUI 1 次側アプリケーションの作業

RUI 1 次側アプリケーションは、要求/応答単位 (request/response unit: RU) レベルで LU-SSCP セッションと PLU-SLU セッションの両方を制御することができ、これらのセッションで SNA RU の送受信を行えます。PU-SSCP セッションは CS/AIX に対して内部であるため、RUI 1 次側アプリケーションはそれにアクセスできません。

RUI 1 次側アプリケーションは RU レベルで動作するため、2 次 LU との間のデータ・フローを大きく制御できます。ただし、このアプリケーションは、送信する SNA メッセージが有効であり、RU レベル・プロトコル (たとえば、ブラケットやチェーンング) が正しく使用されるようにすることについて、通常の LUA アプリケーションよりも大きな責任を負っています。特に、CS/AIX は、RUI 1 次側アプリケーションが送信した RU の妥当性を検証しないことに注意してください。

RUI 1 次側アプリケーションは、次のことを行います。

- RUI_INIT_PRIMARY を使用してダウンストリーム LU を初期化し、RUI_TERM を使用してそれらを終了する

- 2 次側アプリケーションの開始および停止に合わせて、2 次 LU からの NOTIFY メッセージを処理する
- INIT-SELF および TERM-SELF を処理して、PLU-SLU セッションをアクティブにしたり、非アクティブにする
- データ RU 内で 3270 データ・ストリーム・メッセージを作成、送信、受信、および構文解析する
- RU レベル・プロトコル (要求制御、ブラケット、チェーニング、方向) をインプリメントする
- 暗号化 (必要な場合)
- 圧縮 (必要な場合)

ペーシング

ペーシングは LUA インターフェースによって処理されます。LUA アプリケーションはペーシングを制御する必要はないので、ペーシング・インディケータ・フラグは設定しないでください。

LUA アプリケーションからホストに送られたデータに対してペーシングが使用される場合 (これは BIND で決定されます)、RUI_WRITE verb の完了にしばらく時間がかかることがあります。これは、CS/AIX がさらにデータを送信するには、ホストからのペーシング応答を待たなければならないからです。

LUA アプリケーションを使用して 1 方向 (ホストへ、またはホストから) に大量のデータを転送する場合 (ファイル転送アプリケーションなど)、ホスト設定でその方向でのペーシングの使用を指定する必要があります。これは、データを受信するノードがデータであふれたり、データ・ストレージが不足したりしないようにするためです。

セグメント化

RU のセグメント化は、LUA インターフェースによって処理されます。LUA は、常にすべての RU をアプリケーションに渡します。アプリケーションは、すべての RU を LUA に渡す必要があります。

制約事項

CS/AIX は、RUI 1 次側アプリケーションについて、次のことをサポートしません。

- DLUR をまたがるダウンストリーム PU
- 動的に定義された従属 LU (DDDLU)
- STSN の送信 (シーケンス番号をリセットするには、アプリケーションはセッションを UNBIND してから、再度 BIND する必要があります)。

肯定応答

CS/AIX は、アプリケーションから送信された応答を適切な要求と関連させるために、ホストから受け取った要求の記録を取ります。アプリケーションが応答を送信すると、CS/AIX は、その応答を元の要求のデータと関連させます。その後、その要求に関連付けられていたストレージを解放することができます。

1 次 RUI の SNA 情報

ホストで例外応答のみ (否定応答は送信できるが、肯定応答は送信しない) が指定された場合でも、CS/AIX は、アプリケーションが後で否定応答を送信する場合に備えて、要求の記録を取ります。アプリケーションが応答を送信しないと、その要求に関連付けられているストレージは解放できません。

このため、CS/AIX では、ホストからの例外応答のみという要求に対して、LUA アプリケーションが肯定応答を発行することができるようになっています (これを肯定応答と呼びます)。この応答はホストには送信されず、要求に関連付けられたストレージをクリアするために CS/AIX によって使用されます。

チェーン終了までのデータの除去

ホストが要求単位 (RU) のチェーンを LUA アプリケーションに送信する場合、アプリケーションは、チェーン内の最後の RU を受信してから応答を送信することも、チェーンの最後の RU 以外の RU に対して否定応答を送信することもできます。チェーンの途中で否定応答が送信されると、CS/AIX は後続のすべての RU をそのチェーンから除去し、アプリケーションには送信しません。

CS/AIX は、チェーン内の最後の RU を受信すると、RUI_READ verb または RUI_BID verb の 1 次戻りコードを LUA_NEGATIVE_RSP に設定し、2 次戻りコードを 0 (ゼロ) に設定することによって、そのことをアプリケーションに示します。

ホストは、チェーンの途中で CANCEL などのメッセージを送信することによって、チェーンを終了する場合があります。この場合、RUI_READ verb で CANCEL メッセージがアプリケーションに戻され、LUA_NEGATIVE_RSP 戻りコード (39 ページの『否定応答と SNA センス・コード』を参照) は使用されません。

構成情報

CS/AIX 構成ファイルには、LUA アプリケーションが通信を行うために必要な情報が格納されています。このファイルのセットアップと保守は、システム管理者が行います。詳しくは、「*Communications Server for AIX 管理者用ガイド*」を参照してください。

AIX、LINUX

ダウンストリーム LU と通信する RUI 1 次側アプリケーションの場合、必要な構成はダウンストリーム LU (またはダウンストリーム PU テンプレート) のみです。

ホストと通信する LUA アプリケーションで使用するには、以下のコンポーネントを構成する必要があります。

データ・リンク制御 (DLC)、ポート、およびリンク・ステーション (LS)

CS/AIX がリモート・ホスト・コンピューターとの通信に使用する通信コンポーネント。

LU

タイプ 0 から 3 の LU。LU 番号は、ホスト上の適切な LU の番号と一致します。

LU プール (オプション)

必要であれば、アプリケーションで使用する複数の LU を構成し、それらの LU をまとめて 1 つのプールを作ることができます。プールを作成すると、アプリケーションはセッションを開始するときに個々の LU ではなくプールを指定でき、そのプールの中で最初に使用可能になった LU がアプリケーションに割り当てられます。

LUA アプリケーションは、LU 名を指定して `RUI_INIT verb` または `SLI_OPEN verb` を発行することによって、セッションを開始することを CS/AIX に示します。指定する LU 名は、構成ファイル内のタイプ 0 から 3 の LU の名前、または LU プールの名前と一致していなければなりません。CS/AIX は、この名前を次のように使用します。

- 指定された名前がプールに入っていない LU の名前の場合、その LU が使用可能であれば (つまり、すでに別のプログラムによって使用されていない場合)、その LU を使ってセッションが割り当てられます。
- 指定された名前が LU プールの名前またはプール内の LU の名前の場合、そのプールの中で、指定された名前が使用可能であれば、その LU を使ってセッションが割り当てられます。また、指定された名前が使用できない場合はそのプールの中で最初に使用可能な LU を使ってセッションが割り当てられます。`RUI_INIT verb` または `SLI_OPEN verb` は、実際に割り当てられた LU の名前を戻します (これは、指定された名前とは異なる場合があります)。アプリケーションは、ここで戻された LU 名をその後の LUA verb で使用して、セッションを識別することができます。

AIX または Linux に関する考慮事項

AIX、LINUX

この節では、AIX または Linux コンピューター上で LUA アプリケーションを開発する際に念頭に置いておかなければならない処理上の考慮事項について、簡単に説明します。

LUA ヘッダー・ファイル

LUA アプリケーションで使用するヘッダー・ファイルは、`lua_c.h` です。このファイルには、すべての LUA エントリー・ポイントの定義が含まれています。また、共通インターフェース・ヘッダー・ファイル `values_c.h` も含まれています。これらの 2 つのファイルには、LUA インターフェースでの指定パラメーターおよび戻り

AIX または Linux に関する考慮事項

パラメーターの値に対して定義される、すべての定数が含まれます。ファイル `values_c.h` には、LUA VCB で使用される `AP_UINT16` などのパラメーターの型の定義も含まれています。

これら 2 つのファイルは、`/usr/include/sna` (AIX) または `/opt/ibm/sna/include` (Linux) に保管されます。

マルチプロセスと複数セッション

`RUI_INIT`、`RUI_INIT_PRIMARY`、または `SLI_OPEN` を発行したプロセスが `fork` して子プロセスを作成した場合、その子プロセスは、親プロセスが開始したセッションに対しては `LUA verb` を発行することができません。この `verb` を発行すると、その `verb` は失敗し、戻りコード `LUA_UNSUCCESSFUL` と `LUA_INVALID_PROCESS` が戻されます。ただし、子プロセスが新しく `RUI_INIT`、`RUI_INIT_PRIMARY`、または `SLI_OPEN` を発行して、独自のセッションを取得することはできます。

単一プロセスでは同時に複数の LUA セッションを使用することができます。そのためには、`RUI_INIT verb`、`RUI_INIT_PRIMARY verb`、または `SLI_OPEN verb` を複数回発行します。セッションごとに異なる LU を使用する必要がありますが、複数のセッションが同じプールを使用することはできます。

1 つの LUA アプリケーションの複数のインスタンスを別々のプロセスとして実行することができますが、それぞれ異なる LU を使用する必要があります。これには、実行時に LU 名を指定するメカニズムを用意する方法と、LU プールを使用する方法があります。2 つのプロセスで同じプールを指定した場合、各プロセスにはそのプールから別々の LU が割り振られます。

LUA アプリケーションのコンパイルとリンク

AIX アプリケーション

32 ビット・アプリケーションをコンパイルおよびリンクするには、次のオプションを使用します。

```
-bimport:/usr/lib/sna/lua_r.exp -I  
/usr/include/sna
```

64 ビット・アプリケーションをコンパイルおよびリンクするには、次のオプションを使用します。

```
-bimport:/usr/lib/sna/lua_r64_5.exp -I  
/usr/include/sna
```

Linux アプリケーション

LUA アプリケーションをコンパイルし、リンクする前に、共用ライブラリーが保管されるディレクトリーを指定して、実行時に、アプリケーションが見つけられるようにします。このためには、環境変数 `LD_RUN_PATH` を `/opt/ibm/sna/lib` に、64 ビット・アプリケーションをコンパイルしている場合には `/opt/ibm/sna/lib64` に設定します。

32 ビット・アプリケーションをコンパイルおよびリンクするには、次のオプションを使用します。

```
-I /opt/ibm/sna/include -L  
/opt/ibm/sna/lib -llua -lsna_r -lpthread
```

64 ビット・アプリケーションをコンパイルおよびリンクするには、次のオプションを使用します。

```
-I /opt/ibm/sna/include -L  
/opt/ibm/sna/lib64 -llua -lsna_r -lpthread
```

Windows に関する考慮事項

WINDOWS

この節では、Windows クライアント上で LUA アプリケーションを開発する際に念頭に置いておかなければならない処理上の考慮事項について、簡単に説明します。

複数セッションと複数タスク

1 つのタスクが同時に複数の LUA セッションを使用することができます。そのためには、RUI_INIT verb または SLI_OPEN verb を複数回発行します。セッションごとに異なる LU を使用する必要がありますが、複数のセッションが同じプールを使用することはできません。

1 つの LUA アプリケーションの複数のインスタンスを別々のタスクとして実行することができますが、それぞれ異なる LU を使用する必要があります。これには、LU プールを使用する方法があります。2 つのタスクで同じプールを指定ことができ、そのプールから異なる LU が割り振られます。

LUA プログラムのコンパイルとリンク

この節では、Windows クライアントでの LUA プログラムのコンパイルおよびリンクについて説明します。

構造体パッキングのコンパイラ・オプション

LUA verb の VCB 構造体はパックされません。このパッキング・メソッドを変更するコンパイラ・オプションを使用しないでください。

DWORD パラメーターは *DWORD* 境界、*WORD* パラメーターはワード境界、*BYTE* パラメーターはバイト境界にあります。

ヘッダー・ファイル

Windows LUA アプリケーションに組み込まれるヘッダー・ファイルは、指定された **winlua.h** です。このファイルは、Windows クライアント・ソフトウェアをインストールしたディレクトリ内のサブディレクトリ **/sdk** にインストールされます。

ロード時リンク

ロード時にプログラムを LUA にリンクするには、プログラムを `winsli32.lib` ライブラリーにリンクします。

実行時リンク

実行時にプログラムを LUA にリンクするには、プログラムに次の呼び出しを組み込みます。

- `LoadLibrary` LUA ダイナミック・リンク・ライブラリー `winsli32.dll` をロードする場合。
- `GetProcAddress` 必要な各 LUA エントリー・ポイントを指定する場合 (SLI など)。
- `FreeLibrary` ライブラリーが必要でなくなった場合。

アプリケーションの終了

アプリケーションを終了する必要がある場合は (たとえば、ユーザーが **ALT F4** を押した結果として `WM_CLOSE` メッセージを受信した場合など)、終了の前に、`WinRUICleanup` 関数または `WinSLICleanup` 関数を呼び出します。これが行われないと、Windows LUA ソフトウェアがアプリケーションの終了を検出したときに、可能な限りのクリーンアップが実行されても、アプリケーションは不確定な状態のまま残ってしまいます。



移植可能なアプリケーションの作成

CS/AIX にインプリメントされた LUA は、IBM の OS/2 拡張版にインプリメントされた LUA と互換性をもつように設計されています。ただし、基礎となるオペレーティング・システムが異なるため、この 2 つの実装形態にはいくつかの違いがあります。オペレーティング・システムの違いについては、個々の `verb` の説明で取り上げます。特に異なる点は、次のとおりです。

- `RUI_REINIT` `verb` は、標準 LUA インターフェース仕様の拡張機能です。これは、Windows 上の `Remote API Client` では使用できず、他の LUA インプリメンテーションでも使用できない場合があります。
- 他のシステムにインプリメントされた LUA では、CS/AIX にインプリメントされた LUA では戻らないいくつかの戻りコードが生成されます。また、CS/AIX 用として予約されているパラメーターが使用されることもあります。
- OS/2 および Windows にインプリメントされた LUA は、どの場合も `far` ポインター (`far *`) を使用します。AIX または Linux にインプリメントされた LUA の場合は、`far` ポインターと `near` ポインターという概念はないので、`far` というワードを省略する必要があります。
- `verb` の非同期戻り機能のサポート方法は、オペレーティング・システムによって異なります。あるオペレーティング・システム用に作成した LUA アプリケーションに `verb` の非同期の戻りに関するセクションがある場合、そのアプリケーションを別のオペレーティング・システムに移植するには、そのセクションを作成し直さなければならないことがあります。

- 他のシステムにインプリメントされた LUA では、LU プールがサポートされない場合があります。

以下に、他の環境に移植可能な CS/AIX LUA アプリケーションを作成するためのガイドラインを示します。

- LUA ヘッダー・ファイルを、パス名接頭部を指定せずに組み込みます。これにより、別のファイル・システムをもつ環境でそのアプリケーションを使用することができます。LUA ヘッダー・ファイルを探すには、コンパイラーで `include` オプションを使用します (46 ページの『LUA アプリケーションのコンパイルとリンク』を参照)。
- パラメーター値と戻りコードには、ヘッダー・ファイルに指定された数値ではなく、シンボリック定数名を使用します。これにより、メモリーでの保管方法に関係なく、正しい値を使用できます。
- データ・バッファー内の SNA のセンス・コードにアクセスする場合は、数値ではなくシンボリック定数を使用してください。これにより、使用しているシステムに合った正しいバイト保管順序になります。
- 現在ご使用のオペレーティング・システムに適用できるもの以外の戻りコードの検査を組み込み (たとえば、`switch` ステートメントで「デフォルトの」ケースを使用する)、適切な診断を行います。
- 予約済みとなっているパラメーターはすべて、0 (ゼロ) に設定します。
- `lua_verb_length` パラメーターを、63 ページの『第 4 章 RUI verb』または 115 ページの『第 5 章 SLI verb』の説明に従って設定します。

移植可能なアプリケーションの作成

第 3 章 LUA VCB 構造体

この章では、すべての LUA verb で使用される LUA verb 制御ブロック構造体について詳しく説明します。

多くのパラメーター値に対して、ヘッダー・ファイル **lua_c.h** および **values_c.h** (AIX または Linux オペレーティング・システム) または **winlua.h** (Windows オペレーティング・システム) でシンボリック定数が定義されます。移植性を確保するため、指定パラメーターの値の設定、および戻りパラメーターの値のテストの際には、数値ではなくシンボリック定数を使用してください。ファイル **values_c.h** には、LUA VCB で使用される AP_UINT16 などのパラメーターの型の定義も含まれています。

「予約済み」とされているパラメーターは、常に 0 (ゼロ) に設定してください。

LUA verb 制御ブロック (VCB) の形式

verb 制御ブロックは次の 2 つの部分で構成されています。

- 共通データ構造。すべての verb で使用されます。
- 特定データ構造。以下の verb にのみ使用されます。
 - RUI_BID
 - RUI_INIT の拡張バージョン (AIX または Linux 環境)
 - SLI_BID
 - SLI_OPEN
 - SLI_SEND

VCB 構造体のいくつかの部分の定義、特にビット・フィールドの配列は、オペレーティング・システムによって異なります。わかりやすくするために、ここでは 1 つのバージョンの配列だけを示しますが、ヘッダー・ファイルには両方のバージョンが定義されています。アプリケーションでビット・フィールド内の値を設定またはテストするときは、すべてのバイトに対してビット単位の AND 演算または OR 演算を使用するのではなく、名前で個々のビットにアクセスするようにします。ビット配列に依存しないでください。

AIX, LINUX

このような違いに対応するため、LUA ヘッダー・ファイルには次の情報が格納されています。

- ファイル **/usr/include/sna/svconfig.h** (AIX) または **/opt/ibm/sna/include/svconfig.h** (Linux) の **#include** 文。
- LUA データ構造内のビット・フィールドの型定義。この定義によって、データ構造は正しい形式で保管されます。この定義は、ファイル **svconfig.h** 内の **PUCHARQD** の設定に依存します。

注: LUA VCB には、「予約済み」としてマークされた多くのパラメーターが含まれています。これらの予約済みパラメーターには、CS/AIX ソフトウェアにより内部的に使用されるものがありますが、このバージョンでは使用されなくても将来のバージョンで使用されるものもあります。アプリケーションでは、これらの予約済みパラメーターには決してアクセスしないでください。verb によって使用される他のパラメーターをアプリケーションが設定する前に、VCB の内容全体をゼロに設定して、これらのパラメーターすべてを確実にゼロに設定しておく必要があります。このように設定しておくこと、CS/AIX がその内部使用パラメーターを誤って解釈することはありません。またこれにより、今後の CS/AIX のバージョンで、これらのパラメーターを使って新しい関数を引き続き使用できるようになります。

VCB の内容をゼロに設定するには、次のように `memset` を使用します。

```
memset(vcb, 0, sizeof(vcb));
```

LUA_VERB_RECORD データ構造

```
typedef struct
{
    struct LUA_COMMON common;
    struct LUA_SPECIFIC specific;
} LUA_VERB_RECORD;
```

共通データ構造

AIX, LINUX

```
struct LUA_COMMON
{
    AP_UINT16    lua_verb;                /* Verb Code */
    AP_UINT16    lua_verb_length;        /* Length of Verb Record */
    AP_UINT16    lua_prim_rc;            /* Primary Return Code */
    AP_UINT32    lua_sec_rc;             /* Secondary Return Code */
    AP_UINT16    lua_opcode;             /* Verb Operation Code */
    AP_UINT32    lua_correlator;         /* User Correlation Field */
    unsigned char lua_luname[8];         /* Local LU Name */
    AP_UINT16    lua_extension_list_offset; /* Offset of DLL Extension List */
    AP_UINT16    lua_cobol_offset;       /* Offset of Cobol Extension */
    AP_UINT32    lua_sid;                /* Session ID */
    AP_UINT16    lua_max_length;         /* Receive Buffer Length */
    AP_UINT16    lua_data_length;        /* Data Length */
    char *       lua_data_ptr;           /* Data Buffer Pointer */
    unsigned long lua_post_handle;        /* Posting handle */

    struct LUA_TH {                      /* LUA TH Fields */
        BIT_FIELD_TYPE flags_fid : 4;    /* Format Identification Type 2 */
        BIT_FIELD_TYPE flags_mpf : 2;    /* Segmenting Mapping Field */
        BIT_FIELD_TYPE flags_odai : 1;   /* OAF-DAF Assignor Indicator */
        BIT_FIELD_TYPE flags_efi : 1;    /* Expedited Flow Indicator */
        BIT_FIELD_TYPE          : 8;     /* Reserved Field */
        unsigned char daf;               /* Destination Address Field */
        unsigned char oaf;               /* Originating Address Field */
        unsigned char snf[2];            /* Sequence Number Field */
    } lua_th;
};
```

LUA verb 制御ブロック (VCB) の形式

```

struct LUA_RH {
    BIT_FIELD_TYPE rri : 1; /* LUA RH Fields */
    BIT_FIELD_TYPE ruc : 2; /* Request-Response Indicator */
    BIT_FIELD_TYPE : 1; /* RU Category */
    BIT_FIELD_TYPE fi : 1; /* Reserved Field */
    BIT_FIELD_TYPE sdi : 1; /* Format Indicator */
    BIT_FIELD_TYPE bci : 1; /* Sense Data Included Ind */
    BIT_FIELD_TYPE eci : 1; /* Begin Chain Indicator */
    BIT_FIELD_TYPE : 1; /* End Chain Indicator */

    BIT_FIELD_TYPE dr1i : 1; /* DR 1 Indicator */
    BIT_FIELD_TYPE : 1; /* Reserved Field */
    BIT_FIELD_TYPE dr2i : 1; /* DR 2 Indicator */
    BIT_FIELD_TYPE ri : 1; /* Response Indicator */
    BIT_FIELD_TYPE : 2; /* Reserved Field */
    BIT_FIELD_TYPE qri : 1; /* Queued Response Indicator */
    BIT_FIELD_TYPE pi : 1; /* Pacing Indicator */

    BIT_FIELD_TYPE bbi : 1; /* Begin Bracket Indicator */
    BIT_FIELD_TYPE ebi : 1; /* End Bracket Indicator */
    BIT_FIELD_TYPE cdi : 1; /* Change Direction Indicator */
    BIT_FIELD_TYPE : 1; /* Reserved Field */
    BIT_FIELD_TYPE csi : 1; /* Code Selection Indicator */
    BIT_FIELD_TYPE edi : 1; /* Enciphered Data Indicator */
    BIT_FIELD_TYPE pdi : 1; /* Padded Data Indicator */
    BIT_FIELD_TYPE : 1; /* Reserved Field */
} lua_rh;

struct LUA_FLAG1 {
    BIT_FIELD_TYPE bid_enable : 1; /* LUA_FLAG1 */
    BIT_FIELD_TYPE reserv1 : 1; /* Bid Enabled Indicator */
    BIT_FIELD_TYPE close_abend : 1; /* reserved */
    BIT_FIELD_TYPE nowait : 1; /* Close Immediate Flag */
    BIT_FIELD_TYPE sscp_exp : 1; /* Don't Wait for Data Flag */
    BIT_FIELD_TYPE sscp_norm : 1; /* SSCP expedited flow */
    BIT_FIELD_TYPE lu_exp : 1; /* SSCP normal flow */
    BIT_FIELD_TYPE lu_norm : 1; /* LU expedited flow */
    BIT_FIELD_TYPE : 1; /* lu normal flow */
} lua_flag1;

unsigned char lua_message_type; /* sna message command type */

struct LUA_FLAG2 {
    BIT_FIELD_TYPE bid_enable : 1; /* LUA_FLAG2 */
    BIT_FIELD_TYPE async : 1; /* Bid Enabled Indicator */
    BIT_FIELD_TYPE : 2; /* flags asynchronous verb completion */
    BIT_FIELD_TYPE sscp_exp : 1; /* reserved */
    BIT_FIELD_TYPE sscp_norm : 1; /* SSCP expedited flow */
    BIT_FIELD_TYPE lu_exp : 1; /* SSCP normal flow */
    BIT_FIELD_TYPE lu_norm : 1; /* LU expedited flow */
    BIT_FIELD_TYPE : 1; /* lu normal flow */
} lua_flag2;

unsigned char lua_resv56[7]; /* Reserved Field */
unsigned char lua_encr_decr_option; /* Cryptography Option */
};

```

WINDOWS

```

struct LUA_COMMON
{
    unsigned short lua_verb; /* Verb Code */
    unsigned short lua_verb_length; /* Length of Verb Record */
    unsigned short lua_prim_rc; /* Primary Return Code */
    unsigned long lua_sec_rc; /* Secondary Return Code */
    unsigned short lua_opcode; /* Verb Operation Code */
    unsigned long lua_correlator; /* User Correlation Field */
    unsigned char lua_luname[8]; /* Local LU Name */
    unsigned short lua_extension_list_offset; /* Offset of DLL Extention List*/
    unsigned short lua_cobol_offset; /* Offset of Cobol Extension */
}

```

LUA verb 制御ブロック (VCB) の形式

```

unsigned long lua_sid; /* Session ID */
unsigned short lua_max_length; /* Receive Buffer Length */
unsigned short lua_data_length; /* Data Length */
char far *lua_data_ptr; /* Data Buffer Pointer */
unsigned long lua_post_handle; /* Posting handle */

struct LUA_TH { /* LUA TH Fields */
    unsigned char flags_fid : 4; /* Format Identification Type 2 */
    unsigned char flags_mpf : 2; /* Segmenting Mapping Field */
    unsigned char flags_odai : 1; /* OAF-DAF Assignor Indicator */
    unsigned char flags_efi : 1; /* Expedited Flow Indicator */
    unsigned char : 8; /* Reserved Field */
    unsigned char daf; /* Destination Address Field */
    unsigned char oaf; /* Originating Address Field */
    unsigned char snf[2]; /* Sequence Number Field */
} lua_th;

struct LUA_RH { /* LUA RH Fields */
    unsigned char rri : 1; /* Request-Response Indicator */
    unsigned char ruc : 2; /* RU Category */
    unsigned char : 1; /* Reserved Field */
    unsigned char fi : 1; /* Format Indicator */
    unsigned char sdi : 1; /* Sense Data Included Ind */
    unsigned char bci : 1; /* Begin Chain Indicator */
    unsigned char eci : 1; /* End Chain Indicator */

    unsigned char dr1i : 1; /* DR 1 Indicator */
    unsigned char : 1; /* Reserved Field */
    unsigned char dr2i : 1; /* DR 2 Indicator */
    unsigned char ri : 1; /* Response Indicator */
    unsigned char : 2; /* Reserved Field */
    unsigned char qri : 1; /* Queued Response Indicator */
    unsigned char pi : 1; /* Pacing Indicator */

    unsigned char bbi : 1; /* Begin Bracket Indicator */
    unsigned char ebi : 1; /* End Bracket Indicator */
    unsigned char cdi : 1; /* Change Direction Indicator */
    unsigned char : 1; /* Reserved Field */
    unsigned char csi : 1; /* Code Selection Indicator */
    unsigned char edi : 1; /* Enciphered Data Indicator */
    unsigned char pdi : 1; /* Padded Data Indicator */
    unsigned char : 1; /* Reserved Field */
} lua_rh;

struct LUA_FLAG1 { /* LUA_FLAG1 */
    unsigned char bid_enable : 1; /* Bid Enabled Indicator */
    unsigned char reserv1 : 1; /* reserved */
    unsigned char close_abend : 1; /* Close Immediate Flag */
    unsigned char nowait : 1; /* Don't Wait for Data Flag */
    unsigned char sscp_exp : 1; /* SSCP expedited flow */
    unsigned char sscp_norm : 1; /* SSCP normal flow */
    unsigned char lu_exp : 1; /* LU expedited flow */
    unsigned char lu_norm : 1; /* lu normal flow */
} lua_flag1;

unsigned char lua_message_type; /* sna message command type */

struct LUA_FLAG2 { /* LUA_FLAG2 */
    unsigned char bid_enable : 1; /* Bid Enabled Indicator */
    unsigned char async : 1; /* flags asynchronous verb completion */

    unsigned char : 2; /* reserved */
    unsigned char sscp_exp : 1; /* SSCP expedited flow */
    unsigned char sscp_norm : 1; /* SSCP normal flow */
    unsigned char lu_exp : 1; /* LU expedited flow */
    unsigned char lu_norm : 1; /* lu normal flow */
} lua_flag2;

unsigned char lua_resv56[7]; /* Reserved Field */
unsigned char lua_encr_decr_option; /* Cryptography Option */
};

```

以下に、これらのデータ構造のフィールドについて説明します。

lua_verb

LUA verbであることを示します。

値は次のとおりです。

LUA_VERB_RUI

RUI verb。

LUA_VERB_SLI

SLI verb。

lua_verb_length

verb 制御ブロック (VCB) の長さ。

lua_prim_rc

LUA によって設定される 1 次戻りコード。

lua_sec_rc

LUA によって設定される 2 次戻りコード。

lua_opcode

発行される LUA verb を識別する verb 命令コード。

lua_correlator

この verb を使用しているアプリケーション内の他の処理と関連させるための、4 バイトの相関係数。LUA ではこのパラメーターは使用されません。

lua_luname

LUA セッションで使用される LU 名 (ASCII)。LU 名を指定することも、LU プール名を指定することもできます。詳しくは、70 ページの『RUI_INIT』または 130 ページの『SLI_OPEN』を参照してください。

AIX, LINUX

RUI_INIT_PRIMARY の場合、これは、SNA ゲートウェイで使用できるように構成されたダウンストリーム LU (またはダウンストリーム LU テンプレート) を定義することにより明示的に作成されたダウンストリーム LU) の *dslu_name* パラメーターと同じでなければなりません。

lua_extension_list_offset

このフィールドは予約済みです。

lua_cobol_offset

このフィールドは予約済みです。

lua_sid この verb が発行された LUA セッションのセッション ID。

LUA verb 制御ブロック (VCB) の形式

lua_max_length

データを受け取るために RUI_READ、RUI_INIT_PRIMARY、または SLI_RECEIVE に指定されるバッファの長さ、または RUI_BID に戻された待ち RU の全長。

lua_data_length

送信されるデータの長さ、または受け取ったデータの実際の長さ。

lua_data_ptr

送信されるデータ、またはデータを受け取るためのデータ・バッファを指すポインター。

lua_post_handle

AIX, LINUX

verb が非同期で完了した場合に、完了を通知するために LUA が呼び出すコールバック・ルーチンを指すポインター。

WINDOWS

RUI 関数呼び出しまたは SLI 関数呼び出しで VCB が使用される場合は、このフィールドをイベント・ハンドルに設定してください。WinRUI 関数呼び出しまたは WinSLI 関数呼び出しで VCB が使用されている場合、このフィールドは予約済みです。

lua_th 送受信されるメッセージの TH (伝送ヘッダー) が入っているデータ構造。次のものがあります。

lua_th.flags_fid

形式識別タイプ 2: 4 ビット

lua_th.flags_mpf

セグメント化マッピング・フィールド: 2 ビット

lua_th.flags_odai

起点アドレス・フィールド - 宛先アドレス・フィールド
(OAF-DAF) アサイナー・インディケータ

lua_th.flags_efi

急送フロー・インディケータ

lua_th.daf

DAF (宛先アドレス・フィールド)

lua_th.oaf

OAF (起点アドレス・フィールド)

lua_th.snf

順序番号フィールド

lua_rh 送受信されるメッセージの RH (要求/応答ヘッダー) が入っているデータ構造。次のものがあります。

<i>lua_rh.rrl</i>	要求応答インディケータ
<i>lua_rh.ruc</i>	RU カテゴリ: 2 ビット
<i>lua_rh.fi</i>	形式インディケータ
<i>lua_rh.sdi</i>	センス・データ・インクルード・インディケータ
<i>lua_rh.bci</i>	チェーン開始インディケータ
<i>lua_rh.eci</i>	チェーン終了インディケータ
<i>lua_rh.dr1i</i>	確定応答 1 インディケータ
<i>lua_rh.dr2i</i>	確定応答 2 インディケータ
<i>lua_rh.ri</i>	例外応答インディケータ (要求の場合)、または応答タイプ・インディケータ (応答の場合)
<i>lua_rh.qri</i>	待ち行列応答インディケータ
<i>lua_rh.pi</i>	ペーシング・インディケータ
<i>lua_rh.bbi</i>	ブラケット開始インディケータ
<i>lua_rh.ebi</i>	ブラケット終了インディケータ
<i>lua_rh.cdi</i>	方向転換インディケータ
<i>lua_rh.csi</i>	コード選択インディケータ
<i>lua_rh.edi</i>	暗号化データ・インディケータ
<i>lua_rh.pdi</i>	埋め込みデータ・インディケータ
<i>lua_flag1</i>	アプリケーションによって指定されるメッセージ用のフラグが入っているデータ構造。次のものがあります。
<i>lua_flag1.bid_enable</i>	ビット使用可能インディケータ

LUA verb 制御ブロック (VCB) の形式

lua_flag1.close_abend
即時クローズ・インディケータ

lua_flag1.nowait
データ非待機フラグ

lua_flag1.sscp_exp
SSCP 急送フロー

lua_flag1.sscp_norm
SSCP 通常フロー

lua_flag1.lu_exp
LU 急送フロー

lua_flag1.lu_norm
LU 通常フロー

lua_message_type
RUI_READ verb または SLI_RECEIVE verb によって受信される (あるいは RUI_BID verb または SLI_BID verb に対して示される) SNA メッセージのタイプ

lua_flag2
LUA によって戻されるメッセージ用のフラグが入っているデータ構造

lua_flag2.bid_enable
送信権要求 (BID) 使用可能インディケータ

lua_flag2.async
verb の非同期完了フラグ

lua_flag2.sscp_exp
SSCP 急送フロー

lua_flag2.sscp_norm
SSCP 通常フロー

lua_flag2.lu_exp
LU 急送フロー

lua_flag2.lu_norm
LU 通常フロー

lua_encr_decr_option
暗号化オプション。SLI の場合、このパラメータは予約済みであり、ゼロに設定する必要があります。

特定データ構造

次の verb には特定データ構造が組み込まれます。

- RUI_BID
- RUI_INIT の拡張形式
- SLI_BID
- SLI_OPEN
- SLI_SEND

```

union LUA_SPECIFIC
{
struct SLI_OPEN open;
unsigned char lua_sequence_number[2];
unsigned char lua_peek_data[12];
struct RUI_INIT init;
};

```

AIX, LINUX

```

struct SLI_OPEN
{
unsigned char lua_init_type; /* Type of Session Initiation */
unsigned char lua_session_type; /* How to process host UNBIND */
AP_UINT16 lua_wait; /* Secondary Retry Wait Time */

struct LUA_EXT_ENTRY
{
unsigned char lua_routine_type; /* Extension Routine Type */
unsigned long lua_routine_ptr; /* Ptr to Extension Routine */
} lua_open_extension[MAX_EXTENSIONS];

char reserved[93]; /* Padding */
unsigned char lua_ending_delim; /* Extension List Delimiter */
};

```

WINDOWS

```

struct SLI_OPEN
{
unsigned char lua_init_type; /* Type of Session Initiation */
unsigned char lua_session_type; /* How to process host UNBIND */
AP_UINT16 lua_wait; /* Secondary Retry Wait Time */

struct LUA_EXT_ENTRY
{
unsigned char lua_routine_type; /* Extension Routine Type */
unsigned char lua_module_name[9]; /* Extension DLL module name */
unsigned char lua_procedure_name[33]; /* Procedure name to call */
} lua_open_extension[MAX_EXTENSIONS];

char reserved[93]; /* Padding */
unsigned char lua_ending_delim; /* Extension List Delimiter */
};

```

```

struct RUI_INIT
{
unsigned char rui_init_format;
unsigned char lua_puname[8];
unsigned char lua_lunumber;
unsigned char wait_for_link;
};

```

RUI_BID および SLI_BID の場合、このデータ構造には、次のフィールドが含まれています。

lua_peek_data

読み取りを待っているデータの最初から 12 バイトまで。

LUA verb 制御ブロック (VCB) の形式

AIX、LINUX

RUI_INIT verb の拡張バージョンの場合、このデータ構造には次のフィールドが含まれています。RUI_INIT の拡張バージョンの詳細については、70 ページの『RUI_INIT』を参照してください。

rui_init_format

予約済み。このパラメーターは 0 (ゼロ) に設定する必要があります。

lua_puname

このセッションで使用する LU を所有している ローカル PU の名前。PU 名は、LS の定義または CS/AIX 構成の内部 PU で指定する必要があります。

lua_lunumber

このセッションで使用する LU の LU 番号。これは、*lua_puname* で指定された PU 名に対して設定されている、LU タイプ 0 から 3 の LU 番号と一致していなければなりません。

wait_for_link

発行時点で使用できない LU (基礎となる通信リンクが非アクティブであるため) に対してアプリケーションが RUI_INIT を発行した場合、通常、RUI_INIT verb は失敗します。このパラメーターを 1 に設定し、RUI_INIT が完了する前に、リンクと LU がアクティブになるまで LUA が待機するようにこのデフォルトの振る舞いを変更するか、0 (ゼロ) に設定して、デフォルトの振る舞いが行われるようにします。

WINDOWS

RUI_INIT verb の拡張形式は、Windows には適用されません。RUI_INIT データ構造は使用されません。

SLI_OPEN の場合、このデータ構造には次のフィールドが含まれています。これらのパラメーターの詳細については、130 ページの『SLI_OPEN』を参照してください。

lua_init_type

LUA がセッションを開始する方法を指定します (セッション開始を行うのが 1 次か 2 次か、また、必要な SNA メッセージ・シーケンス)。

lua_session_type

LUA が UNBIND タイプ X'01' (通常) を処理する方法を指定します。通常セッションであるか専用セッションであるかには関係ありません。

lua_wait

2 番目に開始されたセッション起動の再試行タイムアウト (秒数)。

lua_open_extension

アプリケーションの SLI_OPEN 拡張ルーチンがある場合は、その情報を含む構造体。

lua_open_extension.lua_routine_type

拡張ルーチンのタイプ (BIND、SDT、または STSN)。

AIX、LINUX

lua_open_extension.lua_routine_ptr

拡張ルーチンのエントリー・ポイントへのポインター。

WINDOWS

lua_open_extension.lua_module_name

拡張モジュールを含む DLL の名前。

lua_open_extension.lua_procedure_name

拡張モジュール DLL 内で呼び出すプロシージャー名。

■■■■■

lua_ending_delim

CS/AIX SLI インターフェースでは、このパラメーターは使用されません。これは、元は他の SLI インプリメンテーション用に作成されたアプリケーションとの互換性を保持するためのものです。

SLI_SEND の場合、このデータ構造には次のフィールドが含まれています。

lua_sequence_number

LUA がデータ (データに RU のチェーンが必要な場合は最初の RU) の送信に使用する RU の順序番号。これは行形式で保管されます。

LUA verb 制御ブロック (VCB) の形式

第 4 章 RUI verb

この章では、個々の LUA RUI verb について説明します。verb ごとに、次の情報が記載されています。

- verb の目的。
- LUA との間でやり取りされるパラメーター (VCB フィールド)。各パラメーターの説明では、そのパラメーターに有効な値についての情報を示し、必要に応じて追加情報も示します。
- 他の verb との関連。
- verb の使用法に関する追加情報。

すべての verb で使用される verb 制御ブロック (VCB) の詳細については、51 ページの『第 3 章 LUA VCB 構造体』を参照してください。

多くのパラメーター値に対して、ヘッダー・ファイル `lua_c.h` および `values_c.h` (AIX または Linux オペレーティング・システム) または `winlua.h` (Windows オペレーティング・システム) でシンボリック定数が定義されます。移植性を確保するため、指定パラメーターの値の設定、および戻りパラメーターの値のテストの際には、数値ではなくシンボリック定数を使用してください。ファイル `values_c.h` には、LUA VCB で使用される `AP_UINT16` などのパラメーターの型の定義も含まれています。

「予約済み」とされているパラメーターは、常に 0 (ゼロ) に設定してください。

RUI_BID

`RUI_BID verb` は、待機中の受信メッセージをいつ読み取るかを決定するために、アプリケーションが使用します。アプリケーションはこの `verb` を使用することにより、`RUI_READ verb` を発行する前に、どのデータが使用可能か (使用可能なデータがある場合) を判別できます。

使用可能なメッセージがあると、`RUI_BID verb` は、そのメッセージが受信されたメッセージ・フローの詳細、メッセージ・タイプ、そのメッセージの `TH` および `RH`、最大 12 バイトのメッセージ・データを戻します。

`RUI_BID` と `RUI_READ` との主な違いは、`RUI_BID` ではアプリケーションはデータを着信メッセージ・キューから除去せずに検査できるため、そのデータをそのままにして、後でアクセスできる点です。`RUI_READ verb` では、メッセージは待ち行列から除去されるので、アプリケーションはいったんデータを読み取ると、そのデータを処理しなければなりません。

指定パラメーター

アプリケーションが指定するパラメーターは次のとおりです。

```
lua_verb
    LUA_VERB_RUI
```

lua_verb_length

LUA verb レコードの長さ (バイト数)。

これは sizeof (LUA_VERB_RECORD) に設定してください。

lua_opcode

LUA_OPCODE_RUI_BID

lua_correlator

オプション。この verb をアプリケーション内の他の処理と相互に関連付けるために使用できる 4 バイトの値。LUA は、この情報を使用することも、変更することはありません。

lua_luname

セッションで使用する LU の ASCII 文字の名前。これは、(RUI_INIT verb または RUI_INIT_PRIMARY verb で戻された) アクティブな LUA セッションの LU 名と一致していなければなりません。

このパラメーターは、*lua_sid* パラメーターが 0 (ゼロ) の場合にのみ必要です。*lua_sid* でセッション ID が指定されている場合は、LUA はこのパラメーターを使用しません。

このパラメーターの長さは 8 バイトにしてください。名前が 8 文字未満の場合は、右側をスペース (0x20) で埋めてください。

lua_sid セッションのセッション ID。これは、直前の RUI_INIT verb または RUI_INIT_PRIMARY verb で戻されたセッション ID と一致していなければなりません。

このパラメーターはオプションです。セッション ID を指定しない場合は、*lua_luname* パラメーターでセッションの LU 名を指定する必要があります。

lua_post_handle

AIX, LINUX

verb が非同期で完了した場合に、完了を通知するために LUA が呼び出すコールバック・ルーチンを指すポインター。

WINDOWS

RUI 関数呼び出し内で VCB が使用されている場合は、このフィールドをイベント・ハンドルに設定してください。WinRUI 関数呼び出しで VCB が使用されている場合、このフィールドは予約済みです。

■■■■■

詳しくは、13 ページの『第 2 章 LUA アプリケーションの設計と作成』を参照してください。

戻りパラメーター

LUA は常に次のパラメーターを戻します。

lua_flag2.async

このフラグは、*verb* が非同期で完了した場合は 1 に設定され、その *verb* が同期して完了した場合は 0 (ゼロ) に設定されます。

その他の戻りパラメーターは、*verb* が正常に完了したかどうかによって異なります。以下の節を参照してください。

正常な実行

verb が正常に完了した場合、LUA は次のパラメーターを戻します。

lua_prim_rc

LUA_OK

lua_sid この *verb* の発行時に、アプリケーションがセッション ID ではなく *lua_luname* パラメーターを指定した場合は、LUA はセッション ID を戻します。

lua_max_length

待ちメッセージの総バイト数。

lua_data_length

lua_peek_data パラメーターで戻されたデータのバイト数 (0 から 12)。

lua_th 受信メッセージの TH。

lua_rh 受信メッセージの RH。

lua_message_type

受信メッセージのメッセージ・タイプ。これは次のいずれかになります。

LUA_MESSAGE_TYPE_LU_DATA
 LUA_MESSAGE_TYPE_SSCP_DATA
 LUA_MESSAGE_TYPE_RSP
 LUA_MESSAGE_TYPE_BID
 LUA_MESSAGE_TYPE_BIND
 LUA_MESSAGE_TYPE_BIS
 LUA_MESSAGE_TYPE_CANCEL
 LUA_MESSAGE_TYPE_CHASE
 LUA_MESSAGE_TYPE_CLEAR
 LUA_MESSAGE_TYPE_CRV
 LUA_MESSAGE_TYPE_LUSTAT_LU
 LUA_MESSAGE_TYPE_LUSTAT_SSCP
 LUA_MESSAGE_TYPE_QC
 LUA_MESSAGE_TYPE_QEC
 LUA_MESSAGE_TYPE_RELQ
 LUA_MESSAGE_TYPE_RTR
 LUA_MESSAGE_TYPE_SBI

```
LUA_MESSAGE_TYPE_SHUTD
LUA_MESSAGE_TYPE_SIGNAL
LUA_MESSAGE_TYPE_SDT
LUA_MESSAGE_TYPE_STSN
LUA_MESSAGE_TYPE_UNBIND
```

AIX, LINUX

以下の値は、RUI 1 次側アプリケーション (RUI_INIT_PRIMARY を使用してセッションを開始したもの) にのみ戻すことができます。

```
LUA_MESSAGE_TYPE_INIT_SELF
LUA_MESSAGE_TYPE_NOTIFY
LUA_MESSAGE_TYPE_TERM_SELF
```



lua_flag2

次のフラグのどれかが 1 に設定され、データがどのメッセージ・フローで受信されたかが示されます。

```
lua_flag2.sscp_exp
lua_flag2.lu_exp
lua_flag2.sscp_norm
lua_flag2.lu_norm
```

lua_peek_data

メッセージ・データの最初の 12 バイト (メッセージが 12 バイトより短い場合はメッセージ・データ全部)

実行の失敗

verb が正常に完了しなかった場合、LUA は、エラーのタイプを示す 1 次戻りコードと、実行が成功しなかった理由の詳細を示す 2 次戻りコードを戻します。

verb の取り消し: 以下の戻りコードは、別の *verb* によって取り消されたためにその *verb* が正常に完了しなかったことを示します。

```
lua_prim_rc
LUA_CANCELLED
```

lua_sec_rc

LUA_TERMINATED

この *verb* の保留中に RUI_TERM *verb* が発行されました。

パラメーターの検査: 以下の戻りコードは、指定パラメーターに誤りがあったために、その *verb* が正常に完了しなかったことを示します。

lua_prim_rc

LUA_PARAMETER_CHECK

lua_sec_rc

値は次のとおりです。

LUA_BAD_SESSION_ID

lua_sid パラメーターと一致するセッション ID をもつアクティブな LUA LU セッションはありませんでした。

LUA_BID_ALREADY_ENABLED

このセッションでは前の RUI_BID verb が未解決になっているため、RUI_BID verb はリジェクトされました。各セッションでは、同時に複数の RUI_BID を未解決にしておくことはできません。

AIX、LINUX

LUA_INVALID_POST_HANDLE

lua_post_handle パラメーターが、コールバック・ルーチンを指す有効なポインターではありませんでした。

LUA_RESERVED_FIELD_NOT_ZERO

verb レコードの予約フィールド、またはこの verb で使用されないパラメーターが、ゼロ以外の値に設定されていました。

LUA_VERB_LENGTH_INVALID

lua_verb_length パラメーターの値に、この verb に必要な verb レコードの長さより小さい値が指定されていました。

状態の検査: 以下の戻りコードは、verb が発行されたときのセッション状態では、その verb が無効であったことを示します。

lua_prim_rc

LUA_STATE_CHECK

lua_sec_rc

LUA_NO_RUI_SESSION

この verb で指定された LU 名について RUI_INIT verb または RUI_INIT_PRIMARY verb がまだ正常に完了していないか、あるいはセッションに障害が発生しました。

ホストへの否定応答の送信: 以下の戻りコードは、CS/AIX で、ホストから受け取ったデータにエラーが検出されたことを示します。CS/AIX は、受信メッセージを RUI_READ verb でアプリケーションに渡す代わりに、そのメッセージを (それがチェーンに入っている場合はチェーンの残りも) 廃棄して、否定応答をホストに送信します。LUA は、その後の RUI_READ verb または RUI_BID verb で、否定応答が送信されたことをアプリケーションに通知します。

lua_prim_rc

LUA_NEGATIVE_RSP

lua_sec_rc

2 次戻りコードでは、否定応答でホストに送信されたセンス・コードが示されます。戻されたセンス・コード値の解釈については、38 ページの『SNA 情報』を参照してください。

2 次戻りコードが 0 (ゼロ) の場合は、チェーンの途中のメッセージに対する否定応答の RUI_WRITE に続いて、CS/AIX がそのチェーンのすべてのメッセージを受け取って廃棄したことを示します。

その他の状態: 以下の戻りコードは、指定された verb レコードは有効であったが、verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_UNSUCCESSFUL

*lua_sec_rc***LUA_INVALID_PROCESS**

この verb を発行したオペレーティング・システムのプロセスが、このセッションの RUI_INIT verb または RUI_INIT_PRIMARY verb を発行したプロセスと同じではありませんでした。セッションで verb を発行できるのは、そのセッションを開始したプロセスだけです。

以下の戻りコードは、verb がその他の理由で正常に完了しなかったことを示します。

*lua_prim_rc***LUA_COMM_SUBSYSTEM_ABENDED**

CS/AIX の必須ソフトウェア・コンポーネント (ノードなど) が停止しています。必要であれば、システム管理者に連絡してください。

*lua_prim_rc***LUA_SESSION_FAILURE**

LUA セッションに障害が発生しました。

セッションが RUI_INIT (RUI_INIT_PRIMARY ではありません) を使用して開始されており、2 次戻りコードが LUA_RUI_LOGIC_ERROR でない場合は、RUI_REINIT を使ってこの LU を再初期化できます。再初期化しない場合は、同じ LU に対して RUI_TERM を発行してから RUI_INIT または RUI_INIT_PRIMARY を発行する必要があります。

lua_sec_rc

値は次のとおりです。

LUA_LU_COMPONENT_DISCONNECTED

この戻りコードは、通信リンクまたはホスト LU の問題のために LUA セッションに障害が発生したことを示します。

LUA_RUI_LOGIC_ERROR

この戻りコードは、次のいずれかを示します。

- ホスト・システムが SNA プロトコルに違反している
- LUA 内で内部エラーが検出された

SNA のトレース機能をアクティブにして問題を再現し (必要であればシステム管理者に連絡してください)、ホストが正しいデータを送信しているかどうかを調べてください。それでもまだ問題が解決されない場合は、CS/AIX のサポート担当者に連絡してください。

lua_prim_rc

LUA_INVALID_VERB

lua_verb パラメーターも *lua_opcode* パラメーターも有効ではありませんでした。verb は実行されませんでした。

WINDOWS

lua_prim_rc

LUA_STACK_TOO_SMALL

アプリケーションのスタック・サイズが小さすぎて、LUA が要求を完了できません。アプリケーションのスタック・サイズを大きくしてください。

■■■■■

lua_prim_rc

LUA_COMM_SUBSYSTEM_NOT_LOADED

Remote API Client ソフトウェアが開始されていないか、ノードが始動されていない、あるいは LUA アプリケーション向けに適切に構成されていません。CS/AIX LUA 構成パラメーターを調べて、Remote API Client とノードを始動してから、アプリケーションを実行してください。

lua_prim_rc

LUA_UNEXPECTED_DOS_ERROR

オペレーティング・システムでエラーが発生しました。

lua_sec_rc

この値はオペレーティング・システムからの戻りコードです。ご使用のオペレーティング・システムの資料で、この戻りコードの意味を調べてください。

他の verb との関連

RUI_BID verb を発行するには、RUI_INIT verb または RUI_INIT_PRIMARY verb が正常に完了している必要があります。

各セッションで同時に未解決にしておける RUI_BID は 1 つだけです。

RUI_BID verb が正常に完了すれば、後続の RUI_READ verb で *lua_flag1.bid_enable* パラメーターを設定することにより、RUI_BID verb を再発行することができます。この方法で verb を再発行する場合は、アプリケーション・プログラムでは、その RUI_BID verb レコードに関連付けられたストレージを解放したり変更したりしないでください。

RUI_READ と RUI_BID の両方が未解決のときにホストからメッセージが送信されると、RUI_READ は完了し、RUI_BID は進行中のままになります。

使用法と制約事項

到着する各メッセージの送信権要求は、一度だけ行われます。RUI_BID verb によって、特定のセッション・フローでデータが待機していることが通知されると、アプリケーションは RUI_READ verb を発行してそのデータを受信する必要があります。送信権要求されたメッセージが RUI_READ verb の発行によって受け入れられるまでは、後続の RUI_BID では、そのセッション・フローに到着するデータは報告されません。

この verb で戻される *lua_max_length* パラメーターと *lua_data_length* パラメーターの違いは、次のとおりです。

- *lua_max_length* パラメーターは、待ちメッセージの長さを示します。そのメッセージを受け入れるために RUI_READ verb を発行するときに、このパラメーターで指定されたサイズ以上のデータ・バッファーをアプリケーションで提供して、そのメッセージが切り捨てられることなく、すべてが受信できるようにしてください。
- *lua_data_length* パラメーターは、*lua_peek_data* 内のデータの長さを示します。このパラメーターの値が 12 未満 (待ちメッセージが 12 バイトより短い) の場合は、*lua_peek_data* の残りのバイトは未定義です。アプリケーションでは、未定義のバイトを使わないようにしてください。

RUI_INIT

RUI_INIT verb は、指定された LU の SSCP-LU セッション、または指定された LU プール内で最初に使用可能な LU の SSCP-LU セッションを確立します。

AIX, LINUX

一般に、セッションに使用する LU または LU プールの名前はアプリケーションが指定します。CS/AIX には RUI_INIT の拡張バージョンもあり、この形式では、アプリケーションは LU 名を指定する代わりにその PU 名と LU 番号を指定することによって LU を指定できます。この機能は、他のシステムにインプリメントされた LUA ではサポートされていません。RUI_INIT の標準バージョンと拡張バージョンとの違いについては、この節のパラメーターの説明の中で随時説明します。

RUI アプリケーションがダウンストリーム LU と通信するための 1 次 SNA として機能する場合、RUI_INIT ではなく、RUI_INIT_PRIMARY を使用する必要があります。

指定パラメーター

アプリケーションが指定するパラメーターは次のとおりです。

lua_verb

LUA_VERB_RUI

lua_verb_length

LUA verb レコードの長さ (バイト数)。

AIX, LINUX

これは sizeof (LUA_VERB_RECORD) に設定してください。

RUI_INIT の拡張バージョンではなく標準バージョンを使用している場合は、他の LUA インプリメンテーションとの互換性を保つために、値 sizeof (LUA_COMMON) も指定できます。

WINDOWS

これは sizeof (LUA_COMMON) に設定してください。

■*lua_opcode*

LUA_OPCODE_RUI_INIT

lua_correlator

オプション。この verb をアプリケーション内の他の処理と相互に関連付けるために使用できる 4 バイトの値。LUA は、この情報を使用することも、変更することはありません。

lua_luname

セッションを開始したい LU または LU プールの ASCII 文字の名前。これは、CS/AIX に対して設定された LU タイプ 0 から 3 の LU 名、または LU プールの名前と一致していなければなりません。この名前は、次のように使用されます。

- プールに含まれていない LU の名前を指定した場合、CS/AIX はこの LU を使ってセッションの開始を試みます。アプリケーションは、各 verb ごとにそれぞれ異なる LU を指定した複数の RUI_INIT verb を使用することにより、複数のセッションを開始できます。同じ LU で複数のセッションを開始することはできません。
- LU プールの名前を指定した場合、またはプール内の LU 名を指定した場合、CS/AIX は、指定された LU が使用可能であればその LU を使ってセッションの開始を試みます。それ以外の場合は、そのプール内の最初に使用可能な LU を使ってセッションの開始を試みます。アプリケーションは、同じプールを使って複数のセッションを開始できます。CS/AIX は各セッションに、そのプール内の異なる LU を割り当てます。セッションに使用される実際の LU の名前は、RUI_INIT verb の戻りパラメータで示されます。

このパラメーターの長さは 8 バイトにしてください。名前が 8 文字未満の場合は、右側をスペース (0x20) で埋めてください。

AIX, LINUX

アプリケーションは、RUI_INIT の拡張バージョンを使用して、LU 名ではなく PU 名と LU 番号によって LU を指定できます。その場合は、*lua_luname* パラメーターに 8 つの 2 進ゼロを指定し、*lua_puname* パラメーターと *lua_lunumber* パラメーターに PU 名と LU 番号を指定してください。

lua_post_handle

AIX, LINUX

コールバック・ルーチンを指すポインター。verb が非同期で完了した場合、LUA はこのルーチンを呼び出して verb の完了を通知します。

WINDOWS

RUI 関数呼び出し内で VCB が使用されている場合は、このフィールドをイベント・ハンドルに設定してください。WinRUI 関数呼び出しで VCB が使用されている場合、このフィールドは予約済みです。

詳しくは、13 ページの『第 2 章 LUA アプリケーションの設計と作成』を参照してください。

lua_encr_decr_option

セッション・レベル暗号オプション。CS/AIX は、次の 2 つの値を使用します。

0 セッション・レベル暗号は使用されません。

128 暗号化と暗号解除はアプリケーション・プログラムにより実行されます。

これ以外の値を指定すると、戻りコード `LUA_ENCR_DECR_LOAD_ERROR` が戻されます (1 から 127 の範囲の値は、ユーザー定義の暗号化ルーチンおよび暗号解除ルーチンを示します。これらの値は OS/2 拡張版にインプリメントされた LUA ではサポートされますが、CS/AIX ではサポートされません)。

AIX, LINUX

次に示すパラメーターは、*lua_luname* パラメーターに 8 つの 2 進ゼロが指定された場合 (RUI_INIT の拡張バージョンの場合) にのみ使用されます。*lua_luname* に LU 名を指定する場合 (RUI_INIT の標準バージョン) は、これらのパラメーターは予約済みです。

lua_puname

セッションに使用される LU を所有している PU の名前。この名前は ASCII 文字で、右側にスペース (0x20) を埋め込んでください。この名前は、CS/AIX 設定で定義された PU 名と一致していなければなりません。

lua_lunumber

セッションに使用する LU の LU 番号。これは、指定した PU を使用するために設定された、LU タイプ 0 から 3 の LU 番号と一致していなければなりません。

アプリケーションは、各 *verb* ごとにそれぞれ異なる LU を指定した複数の RUI_INIT *verb* を使用することにより、複数のセッションを開始できます。同じ LU で複数のセッションを開始することはできません。

wait_for_link

発行時点で使用できない LU (基礎となる通信リンクが非アクティブであるため) に対してアプリケーションが RUI_INIT を発行した場合、通常、RUI_INIT *verb* は失敗します。このパラメーターを 1 に設定し、RUI_INIT が完了する前に、リンクと LU がアクティブになるまで LUA が待機するようにこのデフォルトの振る舞いを変更するか、0 (ゼロ) に設定して、デフォルトの振る舞いが行われるようにします。

戻りパラメーター

LUA は常に次のパラメーターを戻します。

lua_flag2.async

このフラグは、*verb* が非同期で完了した場合は 1 に設定され、同期して完了した場合は 0 に設定されます (RUI_INIT は、LUA_PARAMETER_CHECK などのエラーを戻す場合を除いて、常に非同期で完了します)。

その他の戻りパラメーターは、*verb* が正常に完了したかどうかによって異なります。以下の節を参照してください。

正常な実行

verb が正常に実行された場合、LUA は次のパラメーターを戻します。

lua_prim_rc

LUA_OK

lua_sid 新しいセッションのセッション ID。その後の *verb* でそのセッションを指定するために使用できます。

lua_luname

新しいセッションで使用される LU の名前。要求パラメーターの LU 名に LU プールが指定されていた場合、または、アプリケーションが RUI_INIT

の拡張バージョンを使用して、LU 名ではなく PU 名と LU 番号を指定した場合、CS/AIX はこのパラメーターを使って、そのセッションに割り当てられた実際の LU の名前を戻します。その後の `verb` でそのセッションを識別するときは、(要求パラメーターで指定した名前ではなく) このパラメーターで戻された名前を使用する必要があります。

実行の失敗

`verb` が正常に完了しなかった場合、LUA は、エラーのタイプを示す 1 次戻りコードと、実行が成功しなかった理由の詳細を示す 2 次戻りコードを戻します。

verb の取り消し: 以下の戻りコードは、別の `verb` によって取り消されたためにその `verb` が正常に完了しなかったことを示します。

lua_prim_rc
LUA_CANCELLED

lua_sec_rc
LUA_TERMINATED
RUI_INIT が完了する前に RUI_TERM `verb` が発行されました。

パラメーターの検査: 以下の戻りコードは、指定パラメーターに誤りがあったために、その `verb` が正常に完了しなかったことを示します。

lua_prim_rc
LUA_PARAMETER_CHECK

lua_sec_rc
値は次のとおりです。

LUA_INVALID_LUNAME

lua_luname パラメーターで識別される LU がアクティブなノード上で見つかりませんでした。その LU 名または LU プール名が構成ファイルで定義され、それらが構成されているノードが始動されているかを確認してください。

AIX, LINUX

LUA_INVALID_POST_HANDLE

lua_post_handle パラメーターが、コールバック・ルーチンを指す有効なポインターではありませんでした。

LUA_RESERVED_FIELD_NOT_ZERO

`verb` レコードの予約フィールド、またはこの `verb` で使用されないパラメーターが、ゼロ以外の値に設定されていました。

LUA_VERB_LENGTH_INVALID

lua_verb_length パラメーターの値に、この `verb` に必要な `verb` レコードの長さより小さい値が指定されていました。

AIX, LINUX

次に示すパラメーターは、*lua_luname* パラメーターに 8 つの 2 進ゼロが指定された場合 (RUI_INIT の拡張バージョンの場合) にのみ使用されます。*lua_luname* に LU 名を指定する場合 (RUI_INIT の標準バージョン) は、これらのパラメーターは予約済みです。

LUA_INVALID_FORMAT

予約パラメーター *rui_init_format* がゼロ以外の値に設定されていました。

LUA_INVALID_PUNAME

lua_puname パラメーターが、CS/AIX 設定で定義されたどの PU 名とも一致していません。

LUA_INVALID_LUNUMBER

lua_lunumber パラメーターが、指定された PU を使用するように定義された、LU タイプ 0 から 3 の LU の番号と一致していません。

状態の検査: 以下の戻りコードは、*verb* が発行されたときのセッション状態では、その *verb* が無効であったことを示します。

lua_prim_rc
LUA_STATE_CHECK

lua_sec_rc

LUA_DUPLICATE_RUI_INIT

このセッションに対して RUI_INIT *verb* を現在処理中です。

その他の状態: 以下の戻りコードは、指定された *verb* レコードは有効であったが、*verb* が正常に完了しなかったことを示します。

lua_prim_rc
LUA_UNSUCCESSFUL

lua_sec_rc
値は次のとおりです。

LUA_COMMAND_COUNT_ERROR

LU プールの名前、またはプール内の LU の名前を *verb* で指定しましたが、そのプール内の LU はすべて使用中です。

LUA_ENCR_DECR_LOAD_ERROR

その *verb* では、*lua_encr_decr_option* に 0 または 128 以外の値が指定されています。

LUA_INVALID_PROCESS

lua_luname パラメーターで指定された LU は、別のプロセスで使用されています。

LUA_LINK_NOT_STARTED

ホストへの接続が開始されていません。使用できるリンクでアクティブなものはありません。

(その他の値)

その他の 2 次戻りコードはすべて、SNA センス・コードです。戻される SNA センス・コードの解釈については、38 ページの『SNA 情報』を参照してください。

次に示すセンス・コード値は CS/AIX に固有の値であり、CS/AIX 構成とホスト構成の間の不一致を示します。

0x10020000

要求された LU を所有している PU の物理装置活動化 (ACTPU) が、ホストから送信されません。

0x10110000

要求された LU の ACTLU が、ホストから送信されません。これは通常、その LU がホストで設定されていないことを示します。

0x10120000

要求された LU の ACTLU が、ホストから送信されません。ホストは従属 LU の動的定義 (DDDLU) をサポートしていますが、その LU の DDDLU 処理は失敗しました。

以下の戻りコードは、verb がその他の理由で正常に完了しなかったことを示します。

lua_prim_rc

LUA_COMM_SUBSYSTEM_NOT_LOADED

この戻りコードは、次のいずれかの状態を示します。

- Remote API Client ソフトウェアが開始されていません。アプリケーションを実行する前に、Remote API Client を開始してください。
- アクティブな CS/AIX ノードがありません。LUA verb を使用するには、要求された LU を所有しているローカル・ノードまたは要求された LU プール内の 1 つ以上の LU を所有しているローカル・ノードを始動しておく必要があります。必要であれば、システム管理者に連絡してください。

lua_prim_rc

LUA_COMM_SUBSYSTEM_ABENDED

CS/AIX の必須ソフトウェア構成要素 (ノードなど) が終了または停止しています。必要であれば、システム管理者に連絡してください。

lua_prim_rc

LUA_SESSION_FAILURE

LUA セッションに障害が発生しました。

2 次戻りコードが `LUA_RUI_LOGIC_ERROR` でない場合、`RUI_REINIT` を使ってこの LU を再初期化できます。再初期化しない場合は、同じ LU に対して `RUI_TERM` を発行してから `RUI_INIT` を発行する必要があります。

lua_sec_rc

LUA_LU_COMPONENT_DISCONNECTED

通信リンクまたはホスト LU に問題が発生したために、LUA セッションが失敗しました。

lua_prim_rc

LUA_INVALID_VERB

lua_verb パラメーターも *lua_opcode* パラメーターも有効ではありませんでした。verb は実行されませんでした。

WINDOWS

lua_prim_rc

LUA_STACK_TOO_SMALL

アプリケーションのスタック・サイズが小さすぎて、LUA が要求を完了できません。アプリケーションのスタック・サイズを大きくしてください。

lua_prim_rc

LUA_UNEXPECTED_DOS_ERROR

オペレーティング・システムでエラーが発生しました。

lua_sec_rc

この値はオペレーティング・システムからの戻りコードです。ご使用のオペレーティング・システムの資料で、この戻りコードの意味を調べてください。

他の verb との関連

この verb は、セッションで発行される最初の LUA verb でなければなりません。

この verb が正常に完了するまでは、そのセッションでは `RUI_TERM` 以外の LUA verb は発行できません (`RUI_TERM` は、保留中の `RUI_INIT` を終了する verb です)。

そのセッションで発行するその他のすべての verb では、この verb で戻される次のパラメーターのいずれかを使ってセッションを指定する必要があります。

- セッション ID。これは、*lua_sid* パラメーターでアプリケーションに戻されます。
- LU 名。これは、*lua_luname* パラメーターでアプリケーションに戻されます。

使用法と制約事項

RUI_INIT verb は、ホストから ACTLU を受け取った後で完了します。この verb は、必要であれば無期限に待機します。RUI_INIT verb を発行する前に ACTLU がすでに受信されている場合、LUA はホストに NOTIFY を送って、LU の使用準備ができていることを通知します。ACTLU も NOTIFY も、LUA アプリケーションには見えません。

RUI_INIT verb が正常に完了すると、そのセッションではセッション開始時に指定された LU が使用されます。RUI_TERM verb が発行されるまでは、(このアプリケーションまたは他のアプリケーションの) その他の LUA セッションではその LU を使用することはできません。

RUI_INIT verb が 1 次戻りコード LUA_IN_PROGRESS で戻った場合は、セッション ID が *lua_sid* パラメーターで戻されます。このセッション ID は、この verb が正常に完了した場合のセッション ID と同じで、RUI_TERM verb で使用して、未解決の RUI_INIT verb を終了することができます。

RUI_INIT_PRIMARY

AIX、LINUX

RUI_INIT_PRIMARY verb は、ダウンストリーム LU と通信している 1 次 SNA アプリケーションのために SSCP-LU セッションを確立します (RUI アプリケーションがホスト LU と通信するための 2 次 SNA として機能する場合、RUI_INIT_PRIMARY ではなく、RUI_INIT を使用する必要があります)。

指定パラメーター

アプリケーションが指定するパラメーターは次のとおりです。

lua_verb

LUA_VERB_RUI

lua_verb_length

LUA verb レコードの長さ (バイト数)。

これは sizeof (LUA_COMMON) に設定してください。

lua_opcode

LUA_OPCODE_RUI_INIT_PRIMARY

lua_correlator

オプション。この verb をアプリケーション内の他の処理と相互に関連付けるために使用できる 4 バイトの値。LUA は、この情報を使用することも、変更することはありません。

lua_luname

セッションを開始したい LU の ASCII 文字の名前。これは、SNA ゲートウェイで使用できるように構成されたダウンストリーム LU またはダウンストリーム LU テンプレートから明示的に作成された LU の名前と同じでなければなりません。

このパラメーターの長さは 8 バイトにしてください。名前が 8 文字未満の場合は、右側をスペース (0x20) で埋めてください。

lua_max_length

ダウンストリーム PU から受け取った ACTLU(+RSP) RU のコピーを受け取るために指定されるバッファの長さ。アプリケーションがこの情報を受け取る必要がない場合は、*lua_data_ptr* パラメーターにヌル・ポインタを指定できます。この場合、アプリケーションがデータ・バッファを指定する必要はありません。

lua_data_ptr

ダウンストリーム PU から受け取った ACTLU(+RSP) RU のコピーを受け取るために指定されるバッファへのポインタ。アプリケーションがこの情報を受け取る必要がない場合は、ヌル・ポインタを指定できます。情報は戻されません。

lua_post_handle

コールバック・ルーチンを指すポインタ。verb が非同期で完了した場合、LUA はこのルーチンを呼び出して verb の完了を通知します。詳しくは、13 ページの『第 2 章 LUA アプリケーションの設計と作成』を参照してください。

lua_encr_decr_option

セッション・レベル暗号オプション。CS/AIX は、次の 2 つの値を使用します。

0 セッション・レベル暗号は使用されません。

128 暗号化と暗号化解除はアプリケーション・プログラムにより実行されます。

これ以外の値を指定すると、戻りコード `LUA_ENCR_DECR_LOAD_ERROR` が戻されます (1 から 127 の範囲の値は、ユーザー定義の暗号化ルーチンおよび暗号化解除ルーチンを示します。これらの値は OS/2 拡張版にインプリメントされた LUA ではサポートされますが、CS/AIX ではサポートされません)。

戻りパラメーター

LUA は常に次のパラメーターを戻します。

lua_flag2.async

このフラグは、verb が非同期で完了した場合は 1 に設定され、同期して完了した場合は 0 に設定されます (RUI_INIT_PRIMARY は、LUA_PARAMETER_CHECK などのエラーを戻す場合を除いて、常に非同期で完了します)。

その他の戻りパラメーターは、verb が正常に完了したかどうかによって異なります。以下の節を参照してください。

正常な実行

verb が正常に実行された場合、LUA は次のパラメーターを戻します。

lua_prim_rc

LUA_OK

RUI_INIT_PRIMARY

lua_sid 新しいセッションのセッション ID。その後の *verb* でそのセッションを指定するために使用できます。

lua_data_length

ダウンストリーム PU から受け取る ACTLU(+RSP) RU の長さ。LUA は、*lua_data_ptr* で指定されたバッファーにデータを格納します。

実行の失敗

verb が正常に完了しなかった場合、LUA は、エラーのタイプを示す 1 次戻りコードと、実行が成功しなかった理由の詳細を示す 2 次戻りコードを戻します。

verb の取り消し: 以下の戻りコードは、別の *verb* によって取り消されたためにその *verb* が正常に完了しなかったことを示します。

lua_prim_rc

LUA_CANCELLED

lua_sec_rc

LUA_TERMINATED

RUI_INIT_PRIMARY が完了する前に RUI_TERM *verb* が発行されました。

パラメーターの検査: 以下の戻りコードは、指定パラメーターに誤りがあったために、その *verb* が正常に完了しなかったことを示します。

lua_prim_rc

LUA_PARAMETER_CHECK

lua_sec_rc

値は次のとおりです。

LUA_INVALID_LUNAME

lua_luname パラメーターで識別される LU がアクティブなノード上で見つかりませんでした。LU 名が構成ファイルで定義され、それが構成されているノードが始動されているか確認してください。

LUA_INVALID_POST_HANDLE

lua_post_handle パラメーターが、コールバック・ルーチンを指す有効なポインターではありませんでした。

LUA_RESERVED_FIELD_NOT_ZERO

verb レコードの予約フィールド、またはこの *verb* で使用されないパラメーターが、ゼロ以外の値に設定されていました。

LUA_VERB_LENGTH_INVALID

lua_verb_length パラメーターの値に、この *verb* に必要な *verb* レコードの長さより小さい値が指定されていました。

状態の検査: 以下の戻りコードは、*verb* が発行されたときのセッション状態では、その *verb* が無効であったことを示します。

lua_prim_rc

LUA_STATE_CHECK

lua_sec_rc

LUA_DUPLICATE_RUI_INIT_PRIMARY

このセッションに対して RUI_INIT_PRIMARY verb を現在処理中です。

その他の状態: 以下の戻りコードは、指定された verb レコードは有効であったが、verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_UNSUCCESSFUL

lua_sec_rc

値は次のとおりです。

LUA_ENCR_DECR_LOAD_ERROR

その verb では、*lua_encr_decr_option* に 0 または 128 以外の値が指定されています。

LUA_INVALID_PROCESS

lua_luname パラメーターで指定された LU は、別のプロセスで使用されています。

(その他の値)

その他の 2 次戻りコードはすべて、SNA センス・コードです。戻される SNA センス・コードの解釈については、38 ページの『SNA 情報』を参照してください。

以下の戻りコードは、verb がその他の理由で正常に完了しなかったことを示します。

lua_prim_rc

LUA_COMM_SUBSYSTEM_NOT_LOADED

この戻りコードは、次のいずれかの状態を示します。

- Remote API Client ソフトウェアが開始されていません。アプリケーションを実行する前に、Remote API Client を開始してください。
- アクティブな CS/AIX ノードがありません。LUA verb を使用するためには、要求されたダウンストリーム LU を所有するローカル・ノードを始動しておく必要があります。必要であれば、システム管理者に連絡してください。

lua_prim_rc

LUA_COMM_SUBSYSTEM_ABENDED

CS/AIX の必須ソフトウェア構成要素 (ノードなど) が終了または停止しています。必要であれば、システム管理者に連絡してください。

lua_prim_rc

LUA_SESSION_FAILURE

LUA セッションに障害が発生しました。

RUI_TERM を発行してからでないと、同じ LU に対して RUI_INIT_PRIMARY をもう 1 つ発行することはできません。

lua_sec_rc

LUA_LU_COMPONENT_DISCONNECTED

通信リンクまたはホスト LU に問題が発生したために、LUA セッションが失敗しました。

lua_prim_rc

LUA_INVALID_VERB

lua_verb パラメーターも *lua_opcode* パラメーターも有効ではありませんでした。verb は実行されませんでした。

lua_prim_rc

LUA_UNEXPECTED_DOS_ERROR

オペレーティング・システムでエラーが発生しました。

lua_sec_rc

この値はオペレーティング・システムからの戻りコードです。ご使用のオペレーティング・システムの資料で、この戻りコードの意味を調べてください。

他の verb との関連

この verb は、セッションで発行される最初の LUA verb でなければなりません。

この verb が正常に完了するまでは、そのセッションでは RUI_TERM 以外の LUA verb は発行できません (RUI_TERM は、保留中の RUI_INIT_PRIMARY を終了する verb です)。

そのセッションで発行するその他のすべての verb では、この verb からの以下のパラメーターのいずれかを使ってセッションを指定する必要があります。

- セッション ID。これは、*lua_sid* パラメーターでアプリケーションに戻されます。
- LU 名。アプリケーションにより *lua_luname* パラメーターに指定されます。

使用法と制約事項

RUI_INIT_PRIMARY verb は、ダウンストリーム LU から ACTLU 肯定応答を受け取った後で完了します。この verb は、必要であれば無期限に待機します。アプリケーションがこの ACTLU 肯定応答の内容を検査する必要がある場合、CS/AIX が受信メッセージの内容を戻すときに使用する RUI_INIT_PRIMARY (*lua_max_length* パラメーターと *lua_data_ptr* パラメーターを使用) にデータ・バッファーを指定すると検査できます。

RUI_INIT_PRIMARY verb が正常に完了すると、そのセッションではセッション開始時に指定された LU が使用されます。RUI_TERM verb が発行されるか、LUA_SESSION_FAILURE 1 次戻りコードが戻されるまでは、(このアプリケーションまたは他のアプリケーションの) その他の LUA セッションではその LU を使用することはできません。

RUI_INIT_PRIMARY verb が 1 次戻りコード LUA_IN_PROGRESS で戻った場合は、セッション ID が *lua_sid* パラメーターを介して戻されます。このセッション ID は、この verb が正常に完了した場合のセッション ID と同じで、RUI_TERM verb で使用して、未解決の RUI_INIT_PRIMARY verb を終了することができます。

RUI_PURGE

RUI_PURGE verb は、直前の RUI_READ を取り消します。RUI_READ は、*lua_flag1.nowait* (即時に戻す) オプションを使用せずに送信された場合、指定されたフローに使用可能なデータがないと、無期限に待機します。RUI_PURGEは、この待機中の verb を強制的に戻るようにします (1 次戻りコード LUA_CANCELLED が戻されます)。

指定パラメーター

アプリケーションが指定するパラメーターは次のとおりです。

lua_verb

LUA_VERB_RUI

lua_verb_length

LUA verb レコードの長さ (バイト数)。

これは sizeof (LUA_COMMON) に設定してください。

lua_opcode

LUA_OPCODE_RUI_PURGE

lua_correlator

オプション。この verb をアプリケーション内の他の処理と相互に関連付けるために使用できる 4 バイトの値。LUA は、この情報を使用することも、変更することはありません。

lua_luname

セッションで使用する LU の ASCII 文字の名前。これは、RUI_INIT verb または RUI_INIT_PRIMARY verb で戻された、アクティブな LUA セッションの LU 名と一致していなければなりません。

このパラメーターは、*lua_sid* パラメーターが 0 (ゼロ) の場合にのみ必要です。*lua_sid* でセッション ID が指定されている場合は、LUA はこのパラメーターを使用しません。

このパラメーターの長さは 8 バイトにしてください。名前が 8 文字未満の場合は、右側をスペース (0x20) で埋めてください。

lua_sid セッションのセッション ID。これは、直前の RUI_INIT verb または RUI_INIT_PRIMARY verb で戻されたセッション ID と一致していなければなりません。

このパラメーターはオプションです。セッション ID を指定しない場合は、*lua_luname* パラメーターでセッションの LU 名を指定する必要があります。

lua_data_ptr

除去する RUI_READ VCB を指すポインター。

lua_post_handle

AIX, LINUX

verb が非同期で完了した場合に、完了を通知するために LUA が呼び出すコールバック・ルーチンを指すポインター。

WINDOWS

RUI 関数呼び出し内で VCB が使用されている場合は、このフィールドをイベント・ハンドルに設定してください。WinRUI 関数呼び出しで VCB が使用されている場合、このフィールドは予約済みです。

詳しくは、13 ページの『第 2 章 LUA アプリケーションの設計と作成』を参照してください。

戻りパラメーター

LUA は常に次のパラメーターを戻します。

lua_flag2.async

このフラグは、*verb* が非同期で完了した場合は 1 に設定され、その *verb* が同期して完了した場合は 0 (ゼロ) に設定されます。

その他の戻りパラメーターは、*verb* が正常に完了したかどうかによって異なります。以下の節を参照してください。

正常な実行

verb が正常に完了した場合は、次のパラメーターが戻されます。

lua_prim_rc

LUA_OK

lua_sid この *verb* の発行時に、アプリケーションがセッション ID ではなく

lua_luname パラメーターを指定した場合は、LUA はセッション ID を戻します。

実行の失敗

verb が正常に完了しなかった場合、LUA は、エラーのタイプを示す 1 次戻りコードと、実行が成功しなかった理由の詳細を示す 2 次戻りコードを戻します。

verb の取り消し: 以下の戻りコードは、別の *verb* によって取り消されたためにその *verb* が正常に完了しなかったことを示します。

lua_prim_rc

LUA_CANCELLED

lua_sec_rc

LUA_TERMINATED

この verb の保留中に RUI_TERM verb が発行されました。

パラメーターの検査: 以下の戻りコードは、指定パラメーターに誤りがあったために、その verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_PARAMETER_CHECK

lua_sec_rc

値は次のとおりです。

LUA_BAD_DATA_PTR

lua_data_ptr パラメーターが 0 (ゼロ) に設定されていました。

LUA_BAD_SESSION_ID

lua_sid パラメーターと一致するセッション ID をもつアクティブな LUA LU セッションはありませんでした。

AIX, LINUX

LUA_INVALID_POST_HANDLE

lua_post_handle パラメーターが、コールバック・ルーチンを指す有効なポインターではありませんでした。

LUA_RESERVED_FIELD_NOT_ZERO

verb レコードの予約フィールド、またはこの verb で使用されないパラメーターが、ゼロ以外の値に設定されていました。

LUA_VERB_LENGTH_INVALID

lua_verb_length パラメーターの値に、この verb に必要な verb レコードの長さより小さい値が指定されていました。

状態の検査: 以下の戻りコードは、verb が発行されたときのセッション状態では、その verb が無効であったことを示します。

lua_prim_rc

LUA_STATE_CHECK

lua_sec_rc

LUA_NO_RUI_SESSION

この verb で指定された LU 名について RUI_INIT verb または RUI_INIT_PRIMARY verb がまだ正常に完了していないか、あるいはセッションに障害が発生しました。

その他の状態: 以下の戻りコードは、指定された verb レコードは有効であったが、verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_UNSUCCESSFUL

lua_sec_rc

値は次のとおりです。

LUA_INVALID_PROCESS

この verb を発行したオペレーティング・システムのプロセスが、このセッションの RUI_INIT verb または RUI_INIT_PRIMARY verb を発行したプロセスと同じではありませんでした。セッションで verb を発行できるのは、そのセッションを開始したプロセスだけです。

LUA_NO_READ_TO_PURGE

lua_data_ptr パラメーターに RUI_READ VCB を指すポインターが指定されていなかったか、RUI_PURGE verb が発行される前に RUI_READ verb が完了しました。

以下の戻りコードは、verb がその他の理由で正常に完了しなかったことを示します。

lua_prim_rc

LUA_COMM_SUBSYSTEM_ABENDED

CS/AIX の必須ソフトウェア構成要素 (ノードなど) が終了または停止しています。必要であれば、システム管理者に連絡してください。

lua_prim_rc

LUA_SESSION_FAILURE

LUA セッションに障害が発生しました。

セッションが RUI_INIT (RUI_INIT_PRIMARY ではありません) を使用して開始されており、2 次戻りコードが LUA_RUI_LOGIC_ERROR でない場合は、RUI_REINIT を使ってこの LU を再初期化できます。再初期化しない場合は、同じ LU に対して RUI_TERM を発行してから RUI_INIT または RUI_INIT_PRIMARY を発行する必要があります。

lua_sec_rc

値は次のとおりです。

LUA_LU_COMPONENT_DISCONNECTED

この戻りコードは、通信リンクまたはホスト LU の問題のために LUA セッションに障害が発生したことを示します。

LUA_RUI_LOGIC_ERROR

この戻りコードは、次のいずれかを示します。

- ホスト・システムが SNA プロトコルに違反している
- LUA 内で内部エラーが検出された

SNA のトレース機能をアクティブにして問題を再現し (必要であればシステム管理者に連絡してください)、ホストが正しいデータを送信しているかどうかを調べてください。これでもまだ問題が解決されない場合は、CS/AIX のサポート担当者に連絡してください。

lua_prim_rc

LUA_INVALID_VERB

lua_verb パラメーターも *lua_opcode* パラメーターも有効ではありませんでした。verb は実行されませんでした。

WINDOWS

lua_prim_rc

LUA_STACK_TOO_SMALL

アプリケーションのスタック・サイズが小さすぎて、LUA が要求を完了できません。アプリケーションのスタック・サイズを大きくしてください。



lua_prim_rc

LUA_UNEXPECTED_DOS_ERROR

オペレーティング・システムでエラーが発生しました。

lua_sec_rc

この値はオペレーティング・システムからの戻りコードです。ご使用のオペレーティング・システムの資料で、この戻りコードの意味を調べてください。

lua_prim_rc

LUA_COMM_SUBSYSTEM_NOT_LOADED

Remote API Client ソフトウェアが開始されていないか、ノードが始動されていない、あるいは LUA アプリケーション向けに適切に構成されていません。CS/AIX LUA 構成パラメーターを調べて、Remote API Client とノードを始動してから、アプリケーションを実行してください。

他の verb との関連

この verb は、RUI_READ がすでに発行されていて、完了が保留されている (すなわち、1 次戻りコードが IN_PROGRESS である) 場合にのみ使用できます。

RUI_READ

RUI_READ verb は、ホストからアプリケーションの LU に送信されたデータまたは状況情報を受け取ります。

データの読み取り元として特定の 1 つのメッセージ・フロー (LU 通常、LU 急送、SSCP 通常、または SSCP 急送) を指定するか、あるいは複数のメッセージ・フローを指定することができます。同じフローを指定しないように、複数の RUI_READ verb を未解決にすることができます。

指定パラメーター

アプリケーションが指定するパラメーターは次のとおりです。

lua_verb

LUA_VERB_RUI

lua_verb_length

LUA verb レコードの長さ (バイト数)。

これは sizeof (LUA_COMMON) に設定してください。

lua_opcode

LUA_OPCODE_RUI_READ

lua_correlator

オプション。この verb をアプリケーション内の他の処理と相互に関連付けるために使用できる 4 バイトの値。LUA は、この情報を使用することも、変更することはありません。

lua_luname

セッションで使用する LU の ASCII 文字の名前。これは、RUI_INIT verb または RUI_INIT_PRIMARY verb で戻された、アクティブな LUA セッションの LU 名と一致していなければなりません。

このパラメーターは、*lua_sid* パラメーターが 0 (ゼロ) の場合にのみ必要です。*lua_sid* でセッション ID が指定されている場合は、LUA はこのパラメーターを使用しません。

このパラメーターの長さは 8 バイトにしてください。名前が 8 文字未満の場合は、右側をスペース (0x20) で埋めてください。

lua_sid セッションのセッション ID。これは、直前の RUI_INIT verb または RUI_INIT_PRIMARY verb で戻されたセッション ID と一致していなければなりません。

このパラメーターはオプションです。セッション ID を指定しない場合は、*lua_luname* パラメーターでセッションの LU 名を指定する必要があります。

lua_max_length

データを受信するために提供されるバッファの長さ。

lua_data_ptr

データを受信するために提供されるバッファを指すポインター。

lua_post_handle

AIX, LINUX

verb が非同期で完了した場合に、完了を通知するために LUA が呼び出すコールバック・ルーチンを指すポインター。

WINDOWS

RUI 関数呼び出し内で VCB が使用されている場合は、このフィールドをイベント・ハンドルに設定してください。WinRUI 関数呼び出しで VCB が使用されている場合、このフィールドは予約済みです。

詳しくは、13 ページの『第 2 章 LUA アプリケーションの設計と作成』を参照してください。

lua_flag1 パラメーター

読み取り可能なデータであるかどうかに関係なく RUI_READ verb をできるだけ早く戻したい場合は、*lua_flag1.nowait* パラメーターを 1 に設定してください。データが使用可能になるのを待ってから verb を戻したい場合は、0 (ゼロ) に設定してください。

注: *lua_flag1.nowait* パラメーターに 1 を設定するということは、verb が同期完了するというものではありません。LUA ライブラリーは、ローカル・ノードと通信して、データが使用可能かそうでないかを判別する必要があります。そのためには通常、アプリケーションのブロッキングを防ぐために、verb は非同期で戻らなければなりません。このパラメーターは、すぐに使用できるデータがない場合、verb はそのことを示すためにできるだけ早く非同期で戻る、ということを意味しています。

最後に実行した RUI_BID verb を再度使用可能にする場合は、*lua_flag1.bid_enable* パラメーターを 1 に設定してください (これは、前と全く同じパラメーターを指定して RUI_BID をもう一度発行するのと同じです)。RUI_BID を再度使用可能にしない場合は、0 (ゼロ) に設定してください。直前の RUI_BID を再度使用可能にすると、もともとその verb に割り振られていた VCB が再利用されます。したがって、その VCB を解放したり、変更したりしないでください (詳しくは、95 ページの『他の verb との関連』を参照してください)。

次のフラグのうちの 1 つ以上を 1 に設定して、どのメッセージ・フローからデータを読み取るかを指定してください。

lua_flag1.sscp_exp

lua_flag1.lu_exp

lua_flag1.sscp_norm

lua_flag1.lu_norm

複数のフラグを設定した場合、使用可能なデータのうちに最も優先順位の高いデータが戻されます。優先順位は、(高い順から) SSCP 急送、LU 急送、SSCP 通常、LU 通常です。*lua_flag2* グループの対応するフラグが設定されて、どのフローからデータが読み取られたかが示されます (『戻りパラメーター』を参照)。

CS/AIX にインプリメントされた LUA では、SSCP 急送フローのデータは戻されません。アプリケーションでは、他のシステムにインプリメントされた LUA との互換性を保つために *sscp_exp* フラグを設定できますが、このフローではデータは戻されません。

戻りパラメーター

LUA は、常に次のパラメーターを戻します。

lua_flag2.async

このパラメーターは、*verb* が非同期で完了した場合は 1 に設定され、その *verb* が同期して完了した場合は 0 に設定されます。

lua_flag2.bid_enable

このパラメーターは、RUI_BID が正常に再度使用可能になった場合は 1 に設定され、RUI_BID が再度使用可能にならなかった場合は 0 に設定されます。

その他の戻りパラメーターは、*verb* が正常に完了したかどうかによって異なります。以下の節を参照してください。

正常な実行またはデータの切り捨て

verb が正常に実行された場合、LUA は次のパラメーターを戻します。

lua_prim_rc

LUA_OK

verb が正常に完了した場合は、次のパラメーターが戻されます。また、指定された *lua_data_length* パラメーターの値が小さすぎたために、データが切り捨てられて *verb* が戻る場合も、これらのパラメーターが戻されます (93 ページの『その他の状態』を参照)。

lua_sid この *verb* の発行時に、アプリケーションがセッション ID ではなく *lua_luname* パラメーターを指定した場合は、LUA はセッション ID を戻します。

lua_data_length

受け取ったデータの長さ。LUA は、*lua_data_ptr* で指定されたバッファにデータを格納します。

lua_th 受信メッセージの送信ヘッダー (TH) の情報。

lua_rh 受信メッセージの要求応答ヘッダー (RH) の情報。

lua_message_type

受信メッセージのメッセージ・タイプ。これは次のいずれかになります。

LUA_MESSAGE_TYPE_LU_DATA

LUA_MESSAGE_TYPE_SSCP_DATA

LUA_MESSAGE_TYPE_RSP

LUA_MESSAGE_TYPE_BID

LUA_MESSAGE_TYPE_BIND

LUA_MESSAGE_TYPE_BIS

LUA_MESSAGE_TYPE_CANCEL

LUA_MESSAGE_TYPE_CHASE

LUA_MESSAGE_TYPE_CLEAR

LUA_MESSAGE_TYPE_CRV

LUA_MESSAGE_TYPE_LUSTAT_LU

LUA_MESSAGE_TYPE_LUSTAT_SSCP

LUA_MESSAGE_TYPE_QC
 LUA_MESSAGE_TYPE_QEC
 LUA_MESSAGE_TYPE_RELQ
 LUA_MESSAGE_TYPE_RTR
 LUA_MESSAGE_TYPE_SBI
 LUA_MESSAGE_TYPE_SHUTD
 LUA_MESSAGE_TYPE_SIGNAL
 LUA_MESSAGE_TYPE_SDT
 LUA_MESSAGE_TYPE_STSN
 LUA_MESSAGE_TYPE_UNBIND

AIX, LINUX

以下の値は、RUI 1 次側アプリケーション (RUI_INIT_PRIMARY を使用してセッションを開始したもの) にのみ戻すことができます。

LUA_MESSAGE_TYPE_INIT_SELF
 LUA_MESSAGE_TYPE_NOTIFY
 LUA_MESSAGE_TYPE_TERM_SELF

lua_flag2 パラメーター

次のフラグのどれかが 1 に設定され、データがどのメッセージ・フローで受信されたかを示します。

lua_flag2.lu_exp

lua_flag2.sscp_norm

lua_flag2.lu_norm

CS/AIX にインプリメントされた LUA では、SSCP 急送フローでデータが戻されることはありません。したがって、*sscp_exp* フラグが設定されることはありません (他のシステムにインプリメントされた LUA によって設定される場合もあります)。

実行の失敗

verb が正常に完了しなかった場合、LUA は、エラーのタイプを示す 1 次戻りコードと、実行が成功しなかった理由の詳細を示す 2 次戻りコードを戻します。

verb の取り消し: 以下の戻りコードは、別の **verb** によって取り消されたか、内部エラーが検出されたために、その **verb** が正常に完了しなかったことを示します。

lua_prim_rc
 LUA_CANCELLED

RUI_READ

lua_sec_rc

値は次のとおりです。

LUA_PURGED

RUI_READ verb は、RUI_PURGE verb によって取り消されています。

LUA_TERMINATED

この verb の保留中に RUI_TERM verb が発行されました。

パラメーターの検査: 以下の戻りコードは、指定パラメーターに誤りがあったために、その verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_PARAMETER_CHECK

lua_sec_rc

値は次のとおりです。

LUA_BAD_DATA_PTR

lua_data_ptr パラメーターに指定された値が有効ではありませんでした。

LUA_BAD_SESSION_ID

lua_sid パラメーターと一致するセッション ID をもつアクティブな LUA LU セッションはありませんでした。

LUA_BID_ALREADY_ENABLED

RUI_BID verb を再度使用可能にするよう、*lua_flag1.bid_enable* パラメーターが設定されましたが、直前の RUI_BID verb がまだ進行中でした。

LUA_DUPLICATE_READ_FLOW

lua_flag1 グループのフロー・フラグで、未解決状態の RUI_READ verb がすでにある 1 つ以上のセッション・フローが指定されました。各セッション・フローで同時に待ち状態にしておける RUI_READ は 1 つだけです。

LUA_INVALID_FLOW

lua_flag1 フロー・フラグが何も設定されていませんでした。これらのフラグのうち 1 つ以上を 1 に設定して、どのフローから読み取るかを指定する必要があります。

AIX, LINUX

LUA_INVALID_POST_HANDLE

lua_post_handle パラメーターが、コールバック・ルーチンを指す有効なポインターではありませんでした。

LUA_NO_PREVIOUS_BID_ENABLED

RUI_BID verb を再度使用可能にするよう、*lua_flag1.bid_enable* パ

ラメーターが設定されましたが、使用可能にできる直前の RUI_BID verb はありませんでした (詳しくは、95 ページの『他の verb との関連』を参照してください)。

LUA_RESERVED_FIELD_NOT_ZERO

verb レコードの予約フィールド、またはこの verb で使用されないパラメーターが、ゼロ以外の値に設定されていました。

LUA_VERB_LENGTH_INVALID

lua_verb_length パラメーターの値に、この verb に必要な verb レコードの長さより小さい値が指定されていました。

状態の検査: 以下の戻りコードは、verb が発行されたときのセッション状態では、その verb が無効であったことを示します。

lua_prim_rc

LUA_STATE_CHECK

lua_sec_rc

LUA_NO_RUI_SESSION

この verb で指定された LU 名について RUI_INIT verb または RUI_INIT_PRIMARY verb がまだ正常に完了していないか、あるいはセッションに障害が発生しました。

ホストへの否定応答の送信: 次に示す 1 次戻りコードは、次の 2 つのケースのいずれかであることを示します。どちらのケースであるかは、2 次戻りコードで区別できます。

- CS/AIX で、ホストから受け取ったデータにエラーが検出されました。CS/AIX は、受信メッセージを RUI_READ verb でアプリケーションに渡す代わりに、そのメッセージを (それがチェーンに入っている場合はチェーンの残りも) 廃棄して、否定応答をホストに送信します。LUA は、その後の RUI_READ verb または RUI_BID verb で、否定応答が送信されたことをアプリケーションに通知します。
- LUA アプリケーションはそれまでに、チェーンの途中のメッセージに対して否定応答を送信していました。CS/AIX はそのチェーン内の後続のメッセージを除去しており、そのチェーンのすべてのメッセージが受信されて除去されたことをこの戻りコードでアプリケーションに報告しています。

lua_prim_rc

LUA_NEGATIVE_RSP

lua_sec_rc

ゼロ以外の 2 次戻りコードは、否定応答でホストに送信されたセンス・コードです。これは、CS/AIX がホスト・データのエラーを検出し、ホストに否定応答を送信したことを示します。戻されるセンス・コードの解釈については、38 ページの『SNA 情報』を参照してください。

2 次戻りコードが 0 (ゼロ) の場合は、チェーンの途中のメッセージに対する否定応答の RUI_WRITE に続いて、CS/AIX がそのチェーンのすべてのメッセージを受け取って廃棄したことを示します。

その他の状態: 以下の戻りコードは、指定された verb レコードは有効であったが、verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_UNSUCCESSFUL

lua_sec_rc

値は次のとおりです。

LUA_DATA_TRUNCATED

lua_data_length パラメーターに、メッセージで受信したデータの実際の長さより小さい値が指定されていました。*lua_data_length* で指定されたバイト数のデータだけが verb に対して戻され、残りのデータは廃棄されました。この 2 次戻りコードが戻された場合は、その他にもパラメーターが戻されます。90 ページの『正常な実行またはデータの切り捨て』を参照してください。

LUA_NO_DATA

lua_flag1.nowait パラメーターが設定され、データを待たずにただちに制御を戻すように指示されました。指定されたセッション・フローには、その時点で使用可能なデータはありませんでした。

LUA_INVALID_PROCESS

この verb を発行したオペレーティング・システムのプロセスが、このセッションの RUI_INIT verb または RUI_INIT_PRIMARY verb を発行したプロセスと同じではありませんでした。セッションで verb を発行できるのは、そのセッションを開始したプロセスだけです。

以下の戻りコードは、verb がその他の理由で正常に完了しなかったことを示します。

lua_prim_rc

LUA_COMM_SUBSYSTEM_ABENDED

CS/AIX の必須ソフトウェア構成要素 (ノードなど) が終了または停止しています。必要であれば、システム管理者に連絡してください。

lua_prim_rc

LUA_SESSION_FAILURE

LUA セッションに障害が発生しました。

セッションが RUI_INIT (RUI_INIT_PRIMARY ではありません) を使用して開始されており、2 次戻りコードが LUA_RUI_LOGIC_ERROR でない場合は、RUI_REINIT を使ってこの LU を再初期化できます。再初期化しない場合は、同じ LU に対して RUI_TERM を発行してから RUI_INIT または RUI_INIT_PRIMARY を発行する必要があります。

lua_sec_rc

値は次のとおりです。

LUA_LU_COMPONENT_DISCONNECTED

この戻りコードは、通信リンクまたはホスト LU の問題のために LUA セッションに障害が発生したことを示します。

LUA_RUI_LOGIC_ERROR

この戻りコードは、次のいずれかを示します。

- ホスト・システムが SNA プロトコルに違反している
- LUA 内で内部エラーが検出された

SNA のトレース機能をアクティブにして問題を再現し (必要であればシステム管理者に連絡してください)、ホストが正しいデータを送信しているかどうかを調べてください。これでもまだ問題が解決されない場合は、CS/AIX のサポート担当者に連絡してください。

lua_prim_rc

LUA_INVALID_VERB

lua_verb パラメーターも *lua_opcode* パラメーターも有効ではありませんでした。verb は実行されませんでした。

WINDOWS

lua_prim_rc

LUA_STACK_TOO_SMALL

アプリケーションのスタック・サイズが小さすぎて、LUA が要求を完了できません。アプリケーションのスタック・サイズを大きくしてください。

■■■■■

lua_prim_rc

LUA_UNEXPECTED_DOS_ERROR

オペレーティング・システムでエラーが発生しました。

lua_sec_rc

この値はオペレーティング・システムからの戻りコードです。ご使用のオペレーティング・システムの資料で、この戻りコードの意味を調べてください。

lua_prim_rc

LUA_COMM_SUBSYSTEM_NOT_LOADED

Remote API Client ソフトウェアが開始されていないか、ノードが始動されていない、あるいは LUA アプリケーション向けに適切に構成されていません。CS/AIX LUA 構成パラメーターを調べて、Remote API Client とノードを始動してから、アプリケーションを実行してください。

他の verb との関連

RUI BID verb を発行するには、RUI_INIT verb または RUI_INIT_PRIMARY verb が正常に完了している必要があります。

既存の RUI_READ が保留状態のときに別の RUI_READ を発行するには、保留中の RUI_READ とは別のセッション・フローを指定することが条件となります。同じセッション・フローに対して複数の RUI_READ を未解決にしておくことはできません。

lua_flag1.bid_enable パラメーターは、次の条件が満たされている場合にのみ使用できます。

- RUI_BID がすでに正常に発行され、完了している
- RUI_BID verb に割り振られたストレージがまだ解放も変更もされていない
- 保留中の RUI_BID が他にない

このパラメーターを使用して直前の RUI_BID を再度使用可能にする場合、RUI_READ のメッセージ・フロー・フラグのうち少なくとも 1 つを設定して、アプリケーションがどのフロー (1 つ以上) でデータを受け取るかを指定する必要があります。受信される最初のデータが、RUI_READ verb によって受け入れられるフロー上にある場合、RUI_READ はそのデータと共に戻りますが、RUI_BID は戻りません。それ以外の場合は、RUI_BID は戻って読み取り対象のデータがあることを示します (RUI_BID はすべてのフロー上のデータを受け入れるので、RUI_READ がデータを受け入れない場合は必ず RUI_BID がデータを受け入れます)。この場合、アプリケーションは該当するフローに対して別の RUI_READ を発行し、データを取得する必要があります。

特定のフロー上のデータを RUI_BID に優先して RUI_READ で処理するのではなく、すべてのフロー上のデータを RUI_BID で処理する場合は、RUI_READ を使用する代わりに、RUI_BID を明示的に再発行して、直前の RUI_BID を再度使用可能にします。

使用法と制約事項

受け取ったデータが *lua_max_length* パラメーターで指定された長さより長い場合は、データは切り捨てられ、*lua_max_length* で指定されたバイト数のデータだけが戻されます。1 次戻りコード `LUA_UNSUCCESSFUL` および 2 次戻りコード `LUA_DATA_TRUNCATED` も戻されます。

RUI_READ verb を使ってメッセージを読み取ると、その着信されたメッセージはメッセージ・キューから除去され、再びアクセスすることはできなくなります (RUI_BID verb を、非破壊読み取りとして使用することができます。すなわち RUI_BID verb を使うと、アプリケーションは使用可能なデータのタイプを調べることができ、データは着信キュー上に残るので、すぐに使用する必要はありません)。

LUA アプリケーションがメッセージであふれないようにするために、1 次と 2 次間のハーフ・セッション (これはホスト構成で指定されます) でページングを使用できます。LUA アプリケーションがメッセージを読み取る速度が遅い場合、CS/AIX はホストに対するページング応答の送信を遅らせて、ホスト側の速度を遅くします。

RUI_REINIT

AIX, LINUX

RUI_REINIT verb は、セッションに障害が発生した後に、SSCP-LU セッションを再確立します。これは、プールの LU を使用していたアプリケーションが、処理を続行するために同じ LU にアクセスする必要がある場合に使用します (アプリケーションは通常、RUI_TERM を発行し、続いて RUI_INIT を再発行することによって、セッションの障害から回復します。ただし、アプリケーションがプール内の LU を使用していた場合は、RUI_INIT を再発行しても、それまで使用していた LU を必ずしも取得するとは限りません)。

この verb を使用して 1 次 RUI セッション (RUI_INIT_PRIMARY を使用して開始されたもの) を再開することはできません。

指定パラメーター

アプリケーションが指定するパラメーターは次のとおりです。

lua_verb

LUA_VERB_RUI

lua_verb_length

LUA verb レコードの長さ (バイト数)。これは sizeof (LUA_COMMON) に設定してください。

lua_opcode

LUA_OPCODE_RUI_REINIT

lua_correlator

オプション。この verb をアプリケーション内の他の処理と相互に関連付けるために使用できる 4 バイトの値。LUA は、この情報を使用することも、変更することもあります。

lua_luname

障害が発生したセッションで使用されていた LU の ASCII 文字の名前。これは、元の RUI_INIT verb で戻された名前と一致していなければなりません (元の RUI_INIT verb で指定した名前と一致している必要は必ずしもありません)。

このパラメーターは、*lua_sid* パラメーターが 0 (ゼロ) の場合にのみ必要です。*lua_sid* でセッション ID が指定されている場合は、LUA はこのパラメーターを使用しません。

このパラメーターの長さは 8 バイトにしてください。名前が 8 文字未満の場合は、右側をスペース (0x20) で埋めてください。

lua_sid セッションのセッション ID。これは、障害が発生したセッションに対して直前に発行された RUI_INIT verb で戻されたセッション ID と一致していなければなりません。

このパラメーターはオプションです。セッション ID を指定しない場合は、*lua_luname* パラメーターでセッションの LU 名を指定する必要があります。

lua_post_handle

コールバック・ルーチンを指すポインター。verb が非同期で完了した場合、LUA はこのルーチン呼び出して verb の完了を通知します。詳しくは、13 ページの『第 2 章 LUA アプリケーションの設計と作成』を参照してください。

戻りパラメーター

LUA は常に次のパラメーターを戻します。

lua_flag2.async

このフラグは、verb が非同期で完了した場合は 1 に設定され、同期して完了した場合は 0 に設定されます (RUI_REINIT は、LUA_PARAMETER_CHECK などのエラーを戻す場合を除いて、常に非同期で完了します)。

その他の戻りパラメーターは、verb が正常に完了したかどうかによって異なります。以下の節を参照してください。

正常な実行

verb が正常に実行された場合、LUA は次のパラメーターを戻します。

lua_prim_rc

LUA_OK

実行の失敗

verb が正常に完了しなかった場合、LUA は、エラーのタイプを示す 1 次戻りコードと、実行が成功しなかった理由の詳細を示す 2 次戻りコードを戻します。

verb の取り消し: 以下の戻りコードは、別の verb によって取り消されたためにその verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_CANCELLED

lua_sec_rc

LUA_TERMINATED

RUI_REINIT が完了する前に、RUI_TERM verb が発行されました。

パラメーターの検査: 以下の戻りコードは、指定パラメーターに誤りがあったために、その verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_PARAMETER_CHECK

lua_sec_rc

値は次のとおりです。

LUA_INVALID_POST_HANDLE

lua_post_handle パラメーターが、コールバック・ルーチンを指す有効なポインターではありませんでした。

LUA_RESERVED_FIELD_NOT_ZERO

verb レコードの予約フィールド、またはこの verb で使用されないパラメーターが、ゼロ以外の値に設定されていました。

LUA_VERB_LENGTH_INVALID

lua_verb_length パラメーターの値に、この verb に必要な verb レコードの長さより小さい値が指定されていました。

状態の検査: 以下の戻りコードは、verb が発行されたときのセッション状態では、その verb が無効であったことを示します。

lua_prim_rc

LUA_STATE_CHECK

lua_sec_rc

値は次のとおりです。

LUA_NO_RUI_SESSION

指定された LU 名またはセッション ID については、RUI_INIT verb はこれまでに正常に完了していません。

LUA_DUPLICATE_RUI_INIT

このセッションに対して RUI_REINIT verb を現在処理中です。

LUA_REINIT_INVALID

このセッションではセッション障害は発生していません。

その他の状態: 以下の戻りコードは、指定された verb レコードは有効であったが、verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_UNSUCCESSFUL

lua_sec_rc

LUA_INVALID_PROCESS

元の RUI_INIT verb は、別のオペレーティング・システム・プロセスから発行されています。

以下の戻りコードは、verb がその他の理由で正常に完了しなかったことを示します。

lua_prim_rc

LUA_COMM_SUBSYSTEM_NOT_LOADED

この戻りコードは、次のいずれかの状態を示します。

- Remote API Client ソフトウェアが開始されていません。アプリケーションを実行する前に、Remote API Client を開始してください。
- アクティブな CS/AIX ノードがありません。LUA verb を使用するには、要求された LU を所有しているローカル・ノードまたは要求された LU プール内の 1 つ以上の LU を所有しているローカル・ノードを始動しておく必要があります。必要であれば、システム管理者に連絡してください。

lua_prim_rc

LUA_COMM_SUBSYSTEM_ABENDED

CS/AIX の必須ソフトウェア構成要素 (ノードなど) が終了または停止しています。必要であれば、システム管理者に連絡してください。

lua_prim_rc

LUA_SESSION_FAILURE

LUA セッションに障害が発生しました。

2 次戻りコードが `LUA_RUI_LOGIC_ERROR` でない場合、`RUI_REINIT` を使ってこの LU を再初期化できます。再初期化しない場合は、同じ LU に対して `RUI_TERM` を発行してから `RUI_INIT` を発行する必要があります。

lua_sec_rc

値は次のとおりです。

LUA_LU_COMPONENT_DISCONNECTED

通信リンクまたはホスト LU に問題が発生したために、LUA セッションが失敗しました。

LUA_RUI_LOGIC_ERROR

この戻りコードは、次のいずれかを示します。

- ホスト・システムが SNA プロトコルに違反している
- LUA 内で内部エラーが検出された

SNA のトレース機能をアクティブにして問題を再現し (必要であればシステム管理者に連絡してください)、ホストが正しいデータを送信しているかどうかを調べてください。それでもまだ問題が解決されない場合は、CS/AIX のサポート担当者に連絡してください。

lua_prim_rc

LUA_INVALID_VERB

lua_verb パラメーターも *lua_opcode* パラメーターも有効ではありませんでした。verb は実行されませんでした。

lua_prim_rc

LUA_UNEXPECTED_DOS_ERROR

オペレーティング・システムでエラーが発生しました。

lua_sec_rc

この値はオペレーティング・システムからの戻りコードです。ご使用のオペレーティング・システムの資料で、この戻りコードの意味を調べてください。

他の verb との関連

この verb を発行できるのは、直前の LUA verb が 1 次戻りコード `LUA_SESSION_FAILURE` と、`LUA_RUI_LOGIC_ERROR` 以外の 2 次戻りコードで戻った場合だけです。

この verb が正常に完了するまでは、そのセッションでは RUI_TERM 以外の LUA verb は発行できません (RUI_TERM は、保留中の RUI_REINIT を終了する verb です)。

使用法と制約事項

RUI_REINIT verb は、ホストから ACTLU を受け取った後で完了します。この verb は、必要であれば無期限に待機します。RUI_REINIT verb を発行する前にすでに ACTLU が受信されている場合は、この verb は 1 次戻りコード LUA_OK で戻ります。

RUI_REINIT verb が正常に完了すると、このセッションではセッション開始時に指定された LU が使用されます。RUI_TERM verb が発行されるか、LUA_SESSION_FAILURE 1 次戻りコードが戻されるまでは、(このアプリケーションまたは他のアプリケーションの) その他の LUA セッションではその LU を使用することはできません。

2 次戻りコードが LUA_RUI_LOGIC_ERROR でない場合、RUI_REINIT を使ってこの LU を再初期化できます。再初期化しない場合は、同じ LU に対して RUI_TERM を発行してから RUI_INIT を発行する必要があります。

再開されたセッションのセッション ID は、障害が発生する前のセッション ID と同じです。RUI_INIT とは異なり、RUI_REINIT はこのセッション ID を戻しません。アプリケーションでは、元の RUI_INIT verb で戻されたセッション ID を使用するか、その LU 名を使ってセッションにアクセスしてください。



RUI_TERM

RUI_TERM verb は、指定された LU の LU セッションと SSCP セッションの両方を終了します。

指定パラメーター

アプリケーションが指定するパラメーターは次のとおりです。

lua_verb

LUA_VERB_RUI

lua_verb_length

LUA verb レコードの長さ (バイト数)。

これは sizeof (LUA_COMMON) に設定してください。

lua_opcode

LUA_OPCODE_RUI_TERM

lua_correlator

オプション。この verb をアプリケーション内の他の処理と相互に関連付けるために使用できる 4 バイトの値。LUA は、この情報を使用することも、変更することはありません。

lua_luname

セッションで使用する LU の ASCII 文字の名前。これは、RUI_INIT verb または RUI_INIT_PRIMARY verb で戻されたアクティブな LUA セッションの LU 名 (または、未解決の RUI_INIT verb、RUI_INIT_PRIMARY verb、または RUI_REINIT verb で指定された LU 名) と一致していなければなりません。

このパラメーターは、*lua_sid* パラメーターが 0 (ゼロ) の場合にのみ必要です。*lua_sid* でセッション ID が指定されている場合は、LUA はこのパラメーターを使用しません。

このパラメーターの長さは 8 バイトにしてください。名前が 8 文字未満の場合は、右側をスペース (0x20) で埋めてください。

lua_sid セッションのセッション ID。これは、直前の RUI_INIT verb または RUI_INIT_PRIMARY verb で戻されたセッション ID と一致していなければなりません。

このパラメーターはオプションです。セッション ID を指定しない場合は、*lua_luname* パラメーターでセッションの LU 名を指定する必要があります。

lua_post_handle

AIX、LINUX

verb が非同期で完了した場合に、完了を通知するために LUA が呼び出すコールバック・ルーチンを指すポインター。

WINDOWS

RUI 関数呼び出し内で VCB が使用されている場合は、このフィールドをイベント・ハンドルに設定してください。WinRUI 関数呼び出しで VCB が使用されている場合、このフィールドは予約済みです。

詳しくは、13 ページの『第 2 章 LUA アプリケーションの設計と作成』を参照してください。

戻りパラメーター

LUA は常に次のパラメーターを戻します。

lua_flag2.async

このフラグは、verb が非同期で完了した場合は 1 に設定され、同期して完了した場合は 0 に設定されます

その他の戻りパラメーターは、verb が正常に完了したかどうかによって異なります。以下の節を参照してください。

正常な実行

verb が正常に実行された場合、LUA は次のパラメーターを戻します。

```
lua_prim_rc
    LUA_OK
```

実行の失敗

verb が正常に完了しなかった場合、LUA は、エラーのタイプを示す 1 次戻りコードと、実行が成功しなかった理由の詳細を示す 2 次戻りコードを戻します。

パラメーターの検査: 以下の戻りコードは、指定パラメーターに誤りがあったために、その verb が正常に完了しなかったことを示します。

```
lua_prim_rc
    LUA_PARAMETER_CHECK
```

```
lua_sec_rc
    値は次のとおりです。
```

LUA_BAD_SESSION_ID

lua_sid パラメーターと一致するセッション ID をもつアクティブな LUA LU セッションはありませんでした。

AIX, LINUX

LUA_INVALID_POST_HANDLE

lua_post_handle パラメーターが、コールバック・ルーチンを指す有効なポインターではありませんでした。

XXXXXXXXXX

LUA_RESERVED_FIELD_NOT_ZERO

verb レコードの予約フィールド、またはこの verb で使用されないパラメーターが、ゼロ以外の値に設定されていました。

LUA_VERB_LENGTH_INVALID

lua_verb_length パラメーターの値に、この verb に必要な verb レコードの長さより小さい値が指定されていました。

状態の検査: 以下の戻りコードは、verb が発行されたときのセッション状態では、その verb が無効であったことを示します。

```
lua_prim_rc
    LUA_STATE_CHECK
```

```
lua_sec_rc
    LUA_NO_RUI_SESSION
```

この verb で指定された LU 名の LUA セッションがないか、あるいはセッションに障害が発生しました。

未解決の RUI_INIT verb、RUI_INIT_PRIMARY verb、または RUI_REINIT verb を取り消すために、その未解決 verb に対して指定された *lua_luname* パラメーターを使用して RUI_TERM verb が

発行されている場合、この戻りコードは、この verb が処理される前に RUI_INIT、RUI_INIT_PRIMARY、または RUI_REINIT が完了していたことを示す場合があります。その verb が異常終了した (そのため、セッションがない) か、lua_luname で指定されたプールの別の LU を使って RUI_INIT が正常に完了した (そのため、指定された LU 名にセッションがない) 可能性があります。

その他の状態: 以下の戻りコードは、指定された verb レコードは有効であったが、verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_UNSUCCESSFUL

lua_sec_rc

値は次のとおりです。

LUA_COMMAND_COUNT_ERROR

verb が発行されたときに、すでに保留状態の RUI_TERM がありました。

LUA_INVALID_PROCESS

この verb を発行したオペレーティング・システムのプロセスが、このセッションの RUI_INIT verb または RUI_INIT_PRIMARY verb を発行したプロセスと同じではありませんでした。セッションで verb を発行できるのは、そのセッションを開始したプロセスだけです。

以下の戻りコードは、verb がその他の理由で正常に完了しなかったことを示します。

lua_prim_rc

LUA_COMM_SUBSYSTEM_ABENDED

CS/AIX の必須ソフトウェア構成要素 (ノードなど) が終了または停止しています。必要であれば、システム管理者に連絡してください。

lua_prim_rc

LUA_SESSION_FAILURE

LUA セッションに障害が発生しました。

lua_sec_rc

値は次のとおりです。

LUA_LU_COMPONENT_DISCONNECTED

この戻りコードは、通信リンクまたはホスト LU の問題のために LUA セッションに障害が発生したことを示します。

LUA_RUI_LOGIC_ERROR

この戻りコードは、次のいずれかを示します。

- ホスト・システムが SNA プロトコルに違反している
- LUA 内で内部エラーが検出された

SNA のトレース機能をアクティブにして問題を再現し (必要であればシステム管理者に連絡してください)、ホストが正しいデータを送

信しているかどうかを調べてください。これでもまだ問題が解決されない場合は、CS/AIX のサポート担当者に連絡してください。

lua_prim_rc

LUA_INVALID_VERB

lua_verb パラメーターも *lua_opcode* パラメーターも有効ではありませんでした。verb は実行されませんでした。

WINDOWS

lua_prim_rc

LUA_STACK_TOO_SMALL

アプリケーションのスタック・サイズが小さすぎて、LUA が要求を完了できません。アプリケーションのスタック・サイズを大きくしてください。

■■■■■

lua_prim_rc

LUA_UNEXPECTED_DOS_ERROR

オペレーティング・システムでエラーが発生しました。

lua_sec_rc

この値はオペレーティング・システムからの戻りコードです。ご使用のオペレーティング・システムの資料で、この戻りコードの意味を調べてください。

lua_prim_rc

LUA_COMM_SUBSYSTEM_NOT_LOADED

Remote API Client ソフトウェアが開始されていないか、ノードが始動されていない、あるいは LUA アプリケーション向けに適切に構成されていません。CS/AIX LUA 構成パラメーターを調べて、Remote API Client とノードを始動してから、アプリケーションを実行してください。

他の verb との関連

この verb は、RUI_INIT verb、RUI_INIT_PRIMARY verb、または RUI_REINIT verb が発行された後であれば (その verb が完了しているかどうかに関係なく)、いつでも発行できます。

RUI_TERM が発行されたときに保留中の他の LUA verb があると、保留中の verb はそれ以上処理されず、1 次戻りコード LUA_CANCELLED を戻して完了します。

この verb が完了した後は、このセッションについて他の LUA verb を発行することはできません。

AIX, LINUX

RUI_TERM

RUI_INIT_PRIMARY を使用してセッションが開始されている場合、CS/AIX は、ダウンストリーム LU に DACTLU を送信してセッションを終了します。RUI_TERM は、DACTLU 応答を待たずに、戻ります。アプリケーションは、RUI_TERM が終了するとすぐに RUI_INIT_PRIMARY を再発行して、ダウンストリーム LU との新しいセッションを開始できます。ただし、CS/AIX は、DACTLU 応答を受け取るまでこの RUI_INIT_PRIMARY を処理できないため、RUI_INIT_PRIMARY が完了するのに時間がかかる場合があります。



RUI_WRITE

RUI_WRITE verb は、LU セッションまたは SSCP セッションを通して、LUA アプリケーションからホストへ SNA 要求単位または応答単位を送信します。

指定パラメーター

アプリケーションが指定するパラメーターは次のとおりです。

lua_verb

LUA_VERB_RUI

lua_verb_length

LUA verb レコードの長さ (バイト数)。

これは sizeof (LUA_COMMON) に設定してください。

lua_opcode

LUA_OPCODE_RUI_WRITE

lua_correlator

オプション。この verb をアプリケーション内の他の処理と相互に関連付けるために使用できる 4 バイトの値。LUA は、この情報を使用することも、変更することはありません。

lua_luname

セッションで使用する LU の ASCII 文字の名前。これは、RUI_INIT verb または RUI_INIT_PRIMARY verb で戻された、アクティブな LUA セッションの LU 名と一致していなければなりません。

このパラメーターは、*lua_sid* パラメーターが 0 (ゼロ) の場合にのみ必要です。*lua_sid* でセッション ID が指定されている場合は、LUA はこのパラメーターを使用しません。

このパラメーターの長さは 8 バイトにしてください。名前が 8 文字未満の場合は、右側をスペース (0x20) で埋めてください。

lua_sid セッションのセッション ID。これは、直前の RUI_INIT verb または RUI_INIT_PRIMARY verb で戻されたセッション ID と一致していなければなりません。

このパラメーターはオプションです。セッション ID を指定しない場合は、*lua_luname* パラメーターでセッションの LU 名を指定する必要があります。

lua_data_length

提供されるデータの長さ。LU 通常フローでデータを送信する場合、最大長はホストから送られた BIND で指定されます。それ以外のすべてのフローでは、最大長は 256 バイトです。

肯定応答を送信する場合は、通常、このパラメーターは 0 (ゼロ) に設定します。LUA は、指定された順序番号に基づいて応答を完了します。BIND または STSN に対する肯定応答の場合は拡張応答を使用できるので、ゼロ以外の値を使用できます。

否定応答を送信する場合は、このパラメーターはデータ・バッファーに提供される SNA センス・コードの長さ (4 バイト) に設定してください。

lua_data_ptr

提供されるデータを格納するバッファーを指すポインター。

要求、またはデータを必要とする肯定応答の場合、バッファーには RU 全体が格納されます。RU の長さは *lua_data_length* で指定してください。

否定応答の場合は、バッファーには SNA センス・コードが格納されます。

lua_post_handle

AIX, LINUX

verb が非同期で完了した場合に、完了を通知するために LUA が呼び出すコールバック・ルーチンを指すポインター。

WINDOWS

RUI 関数呼び出し内で VCB が使用されている場合は、このフィールドをイベント・ハンドルに設定してください。WinRUI 関数呼び出しで VCB が使用されている場合、このフィールドは予約済みです。

詳しくは、13 ページの『第 2 章 LUA アプリケーションの設計と作成』を参照してください。

lua_th.snf

これは、応答を送信する場合にのみ必要です。応答対象の要求の順序番号を指定します。

lua_rh 要求を送信する場合、*lua_rh* ビットの大部分は、送信するメッセージの RH (要求ヘッダー) と対応するように設定します。*lua_rh.pi* と *lua_rh.qri* は設定しないでください。この 2 つは、LUA によって設定されます。

応答を送信する場合は、次の 2 つの *lua_rh* ビットだけが使用されます。その他のビットは 0 (ゼロ) に設定してください。使用する *lua_rh* ビットは次のとおりです。

lua_rh.rrl

応答を示す 1 に設定します。

lua_rh.ri

肯定応答の場合は 0 に、否定応答の場合は 1 に設定します。

lua_flag1 パラメーター

次のフラグのうちのどれかを 1 に設定して、どのメッセージ・フローでデータを送信するかを指定してください。

lua_flag1.lu_exp

lua_flag1.sscp_norm

lua_flag1.lu_norm

これらのフラグのうちの 1 つだけを 1 に設定してください。CS/AIX では、アプリケーションは SSCP 急送フロー (*lua_flag1.sscp_exp* フラグ) でデータを送信することはできません。

戻りパラメーター

LUA は常に次のパラメーターを戻します。

lua_flag2.async

このフラグは、*verb* が非同期で完了した場合は 1 に設定され、その *verb* が同期して完了した場合は 0 (ゼロ) に設定されます。

その他の戻りパラメーターは、*verb* が正常に完了したかどうかによって異なります。以下の節を参照してください。

正常な実行

verb が正常に実行された場合、LUA は次のパラメーターを戻します。

lua_prim_rc

LUA_OK

lua_sid この *verb* の発行時に、アプリケーションがセッション ID ではなく *lua_luname* パラメーターを指定した場合は、LUA はセッション ID を戻します。

lua_th 書き込まれたメッセージの完了 TH (LUA によって指定されたフィールドを含む)。ホストからの応答との相関のために、*lua_th.snf* (順序番号) の値を保管しなければならないことがあります。

lua_rh 書き込まれたメッセージの完了 RH (LUA によって指定されたフィールドを含む)。

lua_flag2 パラメーター

次のフラグのうちのどれかが 1 に設定され、どのメッセージ・フローでデータが送信されたかが示されます。

lua_flag2.lu_exp

lua_flag2.sscp_norm

lua_flag2.lu_norm

CS/AIX にインプリメントされた LUA では、アプリケーションは SSCP 急送フローでデータを送信することはできません。したがって、*sscp_exp* フラグが設定されることはありません (他のシステムにインプリメントされた LUA によって設定されることはあります)。

実行の失敗

verb が正常に完了しなかった場合、LUA は、エラーのタイプを示す 1 次戻りコードと、実行が成功しなかった理由の詳細を示す 2 次戻りコードを戻します。

verb の取り消し: 以下の戻りコードは、別の verb によって取り消されたためにその verb が正常に完了しなかったことを示します。

lua_prim_rc
LUA_CANCELLED

lua_sec_rc

LUA_TERMINATED

このセッションに対して RUI_TERM verb が発行されたため、verb は取り消されました。

パラメーターの検査: 以下の戻りコードは、指定パラメーターに誤りがあったために、その verb が正常に完了しなかったことを示します。

lua_prim_rc
LUA_PARAMETER_CHECK

lua_sec_rc

値は次のとおりです。

LUA_BAD_DATA_PTR

lua_data_ptr パラメーターに指定された値が有効ではありませんでした。

LUA_BAD_SESSION_ID

lua_sid パラメーターと一致するセッション ID をもつアクティブな LUA LU セッションはありませんでした。

LUA_DUPLICATE_WRITE_FLOW

この verb で指定されたセッション・フローには、未解決状態の RUI_WRITE がすでにあります (セッション・フローは、*lua_flag1* フロー・フラグのどれかを 1 に設定することによって指定します)。各セッション・フローで同時に未解決にしておく RUI_WRITE は 1 つだけです。

LUA_INVALID_FLOW

lua_flag1.sscp_exp フロー・フラグが設定され、SSCP 急送フローでのメッセージの送信が指定されました。CS/AIX では、アプリケーションはこのフローでデータを送信することはできません。

AIX, LINUX

LUA_INVALID_POST_HANDLE

lua_post_handle パラメーターが、コールバック・ルーチンを指す有効なポインターではありませんでした。

LUA_MULTIPLE_WRITE_FLOWS

複数の *lua_flag1* フロー・フラグが 1 に設定されました。フロー・フラグのいずれか 1 つだけを 1 に設定して、データを送信するセッション・フローを指定してください。

LUA_REQUIRED_FIELD_MISSING

この戻りコードは、次のいずれかの状態が発生したことを示します。

- *lua_flag1* フロー・フラグが何も設定されていませんでした。フロー・フラグのいずれか 1 つだけを 1 に設定する必要があります。
- 応答を送信するために RUI_WRITE verb が使用されましたが、その応答では、提供されたデータより多くのデータが必要でした。

LUA_RESERVED_FIELD_NOT_ZERO

verb レコードの予約フィールド、またはこの verb で使用されないパラメーターが、ゼロ以外の値に設定されていました。

LUA_VERB_LENGTH_INVALID

lua_verb_length パラメーターの値に、この verb に必要な verb レコードの長さより小さい値が指定されていました。

状態の検査: 以下の戻りコードは、verb が発行されたときのセッション状態では、その verb が無効であったことを示します。

lua_prim_rc

LUA_STATE_CHECK

lua_sec_rc

値は次のとおりです。

LUA_MODE_INCONSISTENCY

RUI_WRITE で送信された SNA メッセージは、その時点では有効ではありませんでした。これは、LU セッションがバインドされる前に、そのセッションでデータを送信しようとしたことが原因です。送信された SNA メッセージの順序を確認してください。

LUA_NO_RUI_SESSION

この verb で指定された LU 名について RUI_INIT verb または RUI_INIT_PRIMARY verb がまだ正常に完了していないか、あるいはセッションに障害が発生しました。

その他の状態: 以下の戻りコードは、指定された verb レコードは有効であったが、verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_UNSUCCESSFUL

lua_sec_rc

値は次のとおりです。

LUA_FUNCTION_NOT_SUPPORTED

この戻りコードは、次のいずれかの状態が発生したことを示します。

- *lua_rh.fi* bit (形式インディケータ) が 1 に設定されましたが、提供された RU の最初のバイトは認識された要求コードではありませんでした。
- *lua_rh.ruc* パラメーター (RU カテゴリ) で、ネットワーク制御 (NC) カテゴリが指定されました。CS/AIX では、アプリケーションはこのカテゴリの要求は送信できません。

LUA_INVALID_PROCESS

この verb を発行したオペレーティング・システムのプロセスが、このセッションの RUI_INIT verb または RUI_INIT_PRIMARY verb を発行したプロセスと同じではありませんでした。セッションで verb を発行できるのは、そのセッションを開始したプロセスだけです。

LUA_INVALID_SESSION_PARAMETERS

アプリケーションは、RUI_WRITE を使用して、ホストから受け取った BIND メッセージに対する肯定応答を送信しました。しかし、CS/AIX ノードは指定された BIND パラメーターを受け入れることができず、ホストに対して否定応答を送信しました。CS/AIX で受け入れられる BIND プロファイルの詳細については、38 ページの『SNA 情報』を参照してください。

LUA_RSP_CORRELATION_ERROR

RUI_WRITE を使って応答を送信するときに、(応答の対象である受信メッセージの順序番号を示す) *lua_th.snf* パラメーターに有効な値が指定されませんでした。

LUA_RU_LENGTH_ERROR

lua_data_length パラメーターに指定された値が有効ではありませんでした。LU 通常フローでデータを送信する場合、最大長はホストから送られた BIND で指定されます。それ以外のすべてのフローでは、最大長は 256 バイトです。

(その他の値)

その他の 2 次戻りコードはすべて、提供された SNA データが無効であったか、あるいは送信できなかったことを示す、SNA センス・コードです。戻される SNA センス・コードの解釈については、38 ページの『SNA 情報』を参照してください。

以下の戻りコードは、verb がその他の理由で正常に完了しなかったことを示します。

lua_prim_rc

LUA_COMM_SUBSYSTEM_ABENDED

CS/AIX の必須ソフトウェア構成要素 (ノードなど) が終了または停止しています。必要であれば、システム管理者に連絡してください。

lua_prim_rc

LUA_SESSION_FAILURE

LUA セッションに障害が発生しました。

セッションが RUI_INIT (RUI_INIT_PRIMARY ではありません) を使用して開始されており、2 次戻りコードが LUA_RUI_LOGIC_ERROR でない場合は、RUI_REINIT を使ってこの LU を再初期化できます。再初期化しない場合は、同じ LU に対して RUI_TERM を発行してから RUI_INIT または RUI_INIT_PRIMARY を発行する必要があります。

lua_sec_rc

値は次のとおりです。

LUA_LU_COMPONENT_DISCONNECTED

この戻りコードは、通信リンクまたはホスト LU の問題のために LUA セッションに障害が発生したことを示します。

LUA_RUI_LOGIC_ERROR

この戻りコードは、次のいずれかを示します。

- ホスト・システムが SNA プロトコルに違反している
- LUA 内で内部エラーが検出された

SNA のトレース機能をアクティブにして問題を再現し (必要であればシステム管理者に連絡してください)、ホストが正しいデータを送信しているかどうかを調べてください。これでもまだ問題が解決されない場合は、CS/AIX のサポート担当者に連絡してください。

lua_prim_rc

LUA_INVALID_VERB

lua_verb パラメーターも *lua_opcode* パラメーターも有効ではありませんでした。verb は実行されませんでした。

WINDOWS

lua_prim_rc

LUA_STACK_TOO_SMALL

アプリケーションのスタック・サイズが小さすぎて、LUA が要求を完了できません。アプリケーションのスタック・サイズを大きくしてください。

lua_prim_rc

LUA_UNEXPECTED_DOS_ERROR

オペレーティング・システムでエラーが発生しました。

lua_sec_rc

この値はオペレーティング・システムからの戻りコードです。ご使用のオペレーティング・システムの資料で、この戻りコードの意味を調べてください。

lua_prim_rc

LUA_COMM_SUBSYSTEM_NOT_LOADED

Remote API Client ソフトウェアが開始されていないか、ノードが始動されていない、あるいは LUA アプリケーション向けに適切に構成されていません。CS/AIX LUA 構成パラメーターを調べて、Remote API Client とノードを始動してから、アプリケーションを実行してください。

他の verb との関連

この verb を発行するには、RUI_INIT verb または RUI_INIT_PRIMARY verb が正常に発行されている必要があります。

既存の RUI_WRITE が保留状態のときに、さらに RUI_WRITE を発行するには、保留中の RUI_WRITE とは別のセッション・フローを指定することが条件となります。すなわち、同じセッション・フローに対して複数の RUI_WRITE を未解決にすることはできません。

RUI_INIT verb または RUI_INIT_PRIMARY verb が正常に完了した後であればいつでも、RUI_WRITE verb を SSCP 通常フローに対して発行できます。LU 急送フローまたは LU 通常フローに対しては、BIND が受信された後にのみ、RUI_WRITE verb を発行できます。その場合、RUI_WRITE verb は BIND で指定されたプロトコルに従う必要があります。

使用法と制約事項

RUI_WRITE の正常終了は、メッセージが正常にデータ・リンクの待ち行列に入ったことを示します。メッセージが正常に送信されたことや、ホストがメッセージを受け入れたことを示すとは限りません。

CS/AIX LU または ホスト LU の処理能力を超える量のデータを LUA アプリケーションが送信しないようにするために、1 次と2 次間のハーフ・セッション (これは BIND で指定されます) でペーシングを使用することができます。その場合、LU 通常フローに対する RUI_WRITE が遅れたり、完了に時間がかかることがあります。

第 5 章 SLI verb

この章では、個々の LUA SLI verb について説明します。verb ごとに、次の情報が記載されています。

- verb の目的。
- LUA との間でやり取りされるパラメーター (VCB フィールド)。各パラメーターの説明では、そのパラメーターに有効な値についての情報を示し、必要に応じて追加情報も示します。
- 他の verb との関連。
- verb の使用法に関する追加情報。

すべての verb で使用される verb 制御ブロック (VCB) の詳細については、51 ページの『第 3 章 LUA VCB 構造体』を参照してください。

多くのパラメーター値に対して、ヘッダー・ファイル `lua_c.h` および `values_c.h` (AIX オペレーティング・システム) または `winlua.h` (Windows オペレーティング・システム) でシンボリック定数が定義されます。移植性を確保するため、指定パラメーターの値の設定、および戻りパラメーターの値のテストの際には、数値ではなくシンボリック定数を使用してください。ファイル `values_c.h` には、LUA VCB で使用される `AP_UINT16` などのパラメーターの型の定義も含まれています。

「予約済み」とされているパラメーターは、常に 0 (ゼロ) に設定してください。

SLI_BID

SLI_BID verb は、待機中の受信メッセージをいつ読み取るかを決定するために、アプリケーションによって使用されます。アプリケーションはこの verb を使うことにより、SLI_RECEIVE verb を発行する前にどのデータが使用可能か (使用可能なデータがある場合) を判別できます。

使用可能なメッセージがあると、SLI_BID verb は、そのメッセージが受信されたメッセージ・フローの詳細、メッセージ・タイプ、そのメッセージの TH および RH、最大 12 バイトのメッセージ・データを戻します。

SLI_BID と SLI_RECEIVE との主な違いは、SLI_BID ではアプリケーションはデータを着信メッセージ・キューから除去せずに検査できるため、そのデータをそのままにして、後でアクセスできる点です。SLI_RECEIVE verb では、メッセージはキューから除去されるので、アプリケーションはいったんデータを読み取ると、そのデータを処理しなければなりません。

指定パラメーター

アプリケーションが指定するパラメーターは次のとおりです。

```
lua_verb
    LUA_VERB_SLI
```

lua_verb_length

LUA verb レコードの長さ (バイト数)。

これは sizeof (LUA_VERB_RECORD) に設定してください。

lua_opcode

LUA_OPCODE_SLI_BID

lua_correlator

オプション。この verb をアプリケーション内の他の処理と相互に関連付けるために使用できる 4 バイトの値。LUA は、この情報を使用することも、変更することはありません。

lua_luname

セッションで使用する LU の ASCII 文字の名前。これは、(SLI_OPEN verb で戻された) アクティブな LUA セッションの LU 名と一致していなければなりません。

このパラメーターは、*lua_sid* パラメーターが 0 (ゼロ) の場合にのみ必要です。*lua_sid* でセッション ID が指定されている場合は、LUA はこのパラメーターを使用しません。

このパラメーターの長さは 8 バイトにしてください。名前が 8 文字未満の場合は、右側をスペース (0x20) で埋めてください。

lua_sid セッションのセッション ID。これは、直前の SLI_OPEN verb で戻されたセッション ID と一致していなければなりません。

このパラメーターはオプションです。セッション ID を指定しない場合は、*lua_luname* パラメーターでセッションの LU 名を指定する必要があります。

lua_post_handle

AIX, LINUX

verb が非同期で完了した場合に、完了を通知するために LUA が呼び出すコールバック・ルーチンを指すポインター。

WINDOWS

SLI 関数呼び出し内で VCB が使用されている場合は、このフィールドをイベント・ハンドルに設定してください。WinSLI 関数呼び出しで VCB が使用されている場合、このフィールドは予約済みです。

詳しくは、13 ページの『第 2 章 LUA アプリケーションの設計と作成』を参照してください。

戻りパラメーター

LUA は常に次のパラメーターを戻します。

lua_flag2.async

このフラグは、*verb* が非同期で完了した場合は 1 に設定され、その *verb* が同期して完了した場合は 0 (ゼロ) に設定されます。

その他の戻りパラメーターは、*verb* が正常に完了したかどうかによって異なります。以下の節を参照してください。

正常な実行

verb が正常に完了した場合、LUA は次のパラメーターを戻します。

lua_prim_rc

LUA_OK

lua_sid この *verb* の発行時に、アプリケーションがセッション ID ではなく

lua_luname パラメーターを指定した場合は、LUA はセッション ID を戻します。

lua_data_length

lua_peek_data パラメーターで戻されたデータのバイト数 (0 から 12)。

lua_th 受信メッセージの TH。

lua_rh 受信メッセージの RH。

lua_message_type

受信メッセージのメッセージ・タイプ。これは次のいずれかです。

LUA_MESSAGE_TYPE_LU_DATA

LUA_MESSAGE_TYPE_SSCP_DATA

LUA_MESSAGE_TYPE_RSP

LUA_MESSAGE_TYPE_BID

LUA_MESSAGE_TYPE_BIND

LUA_MESSAGE_TYPE_BIS

LUA_MESSAGE_TYPE_CANCEL

LUA_MESSAGE_TYPE_CHASE

LUA_MESSAGE_TYPE_LUSTAT_LU

LUA_MESSAGE_TYPE_LUSTAT_SSCP

LUA_MESSAGE_TYPE_QC

LUA_MESSAGE_TYPE_QEC

LUA_MESSAGE_TYPE_RELQ

LUA_MESSAGE_TYPE_RTR

LUA_MESSAGE_TYPE_SBI

LUA_MESSAGE_TYPE_SIGNAL

LUA_MESSAGE_TYPE_STSN

SLI では、アプリケーションの LUA インターフェース拡張ルーチンを使用して、BIND および STSN 要求の受信とこれらへの応答を行います。

lua_flag2

次のフラグのどれかが 1 に設定され、データがどのメッセージ・フローで受信されたかが示されます。

lua_flag2.sscp_exp

lua_flag2.lu_exp

lua_flag2.sscp_norm

lua_flag2.lu_norm

lua_peek_data

メッセージ・データの最初の 12 バイト (メッセージが 12 バイトより短い場合はメッセージ・データ全部)

lua_rh.rrr がオフ (要求単位) で、*lua_rh.sdi* がオン (センス・データが含まれる) である場合は、LUA が、ホストから送信される要求単位を例外要求 (EXR) に変換したことを示します。この場合、*lua_peek_data* のバイト 0 から 3 には例外に関連したセンス・データが含まれ、バイト 4 から 6 には、元の要求単位の先頭 3 バイトまでが含まれます。

正常な実行: 状況情報

verb により、データでなく、LUA 状況情報が戻された場合、LUA は次のパラメータを戻します。

lua_prim_rc

LUA_STATUS

*lua_sec_rc***LUA_READY**

SLI セッションは、追加のコマンドを処理する準備ができています。この状況情報が使用されるのは、直前の LUA_NOT_READY 状況が報告された後か、あるいは、*lua_prim_rc* を LUA_CANCELLED に、*lua_sec_rc* を RECEIVED_UNBIND_HOLD または RECEIVED_UNBIND_NORMAL に設定して、SLI_CLOSE verb が完了した後です。

LUA_NOT_READY

SLI セッションが、次のいずれかの理由で一時的に中断状態になっています。

- CLEAR コマンドを受信した。SDT コマンドを受信すると、セッションは再開します。
- タイプ X'02' (BIND 受信予定) の UNBIND コマンドを受信した。セッションは、BIND、オプションの CRV および STSN、そして SDT などのコマンドを受信するまで中断状態になり、SDT を受信すると再開します。元の SLI_OPEN verb が提供したどのユーザー拡張ルーチンも再度呼び出されます。
- タイプ X'01' (通常) の UNBIND コマンドを受信し、このセッションの SLI_OPEN verb に *lua_session_type* LUA_SESSION_TYPE_DEDICATED が指定されていた。セッションは、BIND、オプションの CRV および STSN、そして SDT などのコ

マンドを受信するまで中断状態になり、SDT を受信すると再開します。元の SLI_OPEN verb が提供したどのユーザー拡張ルーチンも再度呼び出されます。

セッションの再開時に、アプリケーションは、他の SLI_BID または SLI_RECEIVE を発行して、READY 状況を受け取る必要があります。セッション状況が LUA_NOT_READY であっても、SLI_SEND verb と SLI_RECEIVE verb を SSCP 通常フローのデータに対して発行し続けることができます。

LUA_INIT_COMPLETE

アプリケーションは、LUA_OPEN_TYPE_PRIM_SSCP タイプを指定して SLI_OPEN を発行し、すでに発行されていた RUL_INIT verb は完了しています。アプリケーションは、SSCP 通常フローのデータに対して SLI_SEND verb および SLI_RECEIVE verb を発行できるようになりました。

LUA_SESSION_END_REQUESTED

ホストがアプリケーションに対して、セッションのシャットダウンを要求する SHUTD コマンドを送信しました。アプリケーションは、セッションを閉じる準備ができ次第、SLI_CLOSE を発行する必要があります。

lua_sid この verb の発行時に、アプリケーションがセッション ID ではなく *lua_luname* パラメーターを指定した場合は、LUA はセッション ID を戻します。

実行の失敗

verb が正常に完了しなかった場合、LUA は、エラーのタイプを示す 1 次戻りコードと、実行が成功しなかった理由の詳細を示す 2 次戻りコードを戻します。

verb の取り消し: 以下の戻りコードは、別の verb によって取り消されたためにその verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_CANCELLED

lua_sec_rc

LUA_TERMINATED

この verb の保留中に SLI_CLOSE verb が発行されました。

パラメーターの検査: 以下の戻りコードは、指定パラメーターに誤りがあったために、その verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_PARAMETER_CHECK

lua_sec_rc

値は次のとおりです。

LUA_BAD_SESSION_ID

lua_sid パラメーターと一致するセッション ID をもつアクティブな LUA LU セッションはありませんでした。

LUA_INVALID_LUNAME

lua_luname パラメーターで識別される LU がアクティブなノード上で見つかりませんでした。その LU 名または LU プール名が構成ファイルで定義され、それらが構成されているノードが始動されているかを確認してください。

AIX, LINUX

LUA_INVALID_POST_HANDLE

lua_post_handle パラメーターが、コールバック・ルーチンを指す有効なポインターではありませんでした。

LUA_RESERVED_FIELD_NOT_ZERO

verb レコードの予約フィールド、またはこの *verb* で使用されないパラメーターが、ゼロ以外の値に設定されていました。

LUA_VERB_LENGTH_INVALID

lua_verb_length パラメーターの値に、この *verb* に必要な *verb* レコードの長さより小さい値が指定されていました。

状態の検査: 以下の戻りコードは、*verb* が発行されたときのセッション状態では、その *verb* が無効であったことを示します。

lua_prim_rc

LUA_STATE_CHECK

lua_sec_rc

LUA_NO_SLI_SESSION

この *verb* で指定された LU で、SLI_OPEN *verb* がまだ正常に完了していないか、あるいはセッションに障害が発生しました。

LUA_SLI_BID_PENDING

このセッションでは前の SLI_BID *verb* が未解決になっているため、SLI_BID *verb* はリジェクトされました。各セッションでは、同時に複数の SLI_BID を未解決にしておくことはできません。

ホストへの否定応答の送信: 以下の戻りコードは、CS/AIX で、ホストから受け取ったデータにエラーが検出されたことを示します。CS/AIX は、受信メッセージを SLI_RECEIVE *verb* でアプリケーションに渡す代わりに、そのメッセージを (それがチェーンに入っている場合はチェーンの残りも) 廃棄して、否定応答をホストに送信します。LUA は、その後の SLI_RECEIVE *verb* または SLI_BID *verb* で、否定応答が送信されたことをアプリケーションに通知します。

lua_prim_rc

LUA_NEGATIVE_RSP

lua_sec_rc

2 次戻りコードでは、否定応答でホストに送信されたセンス・コードが示されます。戻されたセンス・コード値の解釈については、38 ページの『SNA 情報』を参照してください。

2 次戻りコードが 0 (ゼロ) の場合は、チェーンの途中のメッセージに対する否定応答の SLI_SEND に続いて、CS/AIX がそのチェーンのすべてのメッセージを受け取って廃棄したことを示します。

その他の状態: 以下の戻りコードは、指定された verb レコードは有効であったが、verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_UNSUCCESSFUL

lua_sec_rc

LUA_INVALID_PROCESS

この verb を発行したオペレーティング・システムのプロセスが、このセッションの SLI_OPEN verb を発行したプロセスと同じではありませんでした。セッションで verb を発行できるのは、そのセッションを開始したプロセスだけです。

以下の戻りコードは、verb がその他の理由で正常に完了しなかったことを示します。

lua_prim_rc

LUA_COMM_SUBSYSTEM_ABENDED

CS/AIX の必須ソフトウェア・コンポーネント (ノードなど) が停止しています。必要であれば、システム管理者に連絡してください。

lua_prim_rc

LUA_SESSION_FAILURE

LUA セッションに障害が発生しました。アプリケーションは、SLI_OPEN を再発行することによって、セッションを再開できません。

lua_sec_rc

値は次のとおりです。

LUA_LU_COMPONENT_DISCONNECTED

この戻りコードは、通信リンクまたはホスト LU の問題のために LUA セッションに障害が発生したことを示します。

LUA_RECEIVED_UNBIND

この戻りコードは、UNBIND コマンドがホストから送信され、セッションが終了したことを示します。この値は、セッションに対する SLI_OPEN verb に *lua_session_type* LUA_SESSION_TYPE_DEDICATED が指定されている場合にのみ生成されます。

LUA_RUI_LOGIC_ERROR

この戻りコードは、次のいずれかを示します。

- ホスト・システムが SNA プロトコルに違反している
- LUA 内で内部エラーが検出された

SNA のトレース機能をアクティブにして問題を再現し (必要であればシステム管理者に連絡してください)、ホストが正しいデータを送信しているかどうかを調べてください。それでもまだ問題が解決されない場合は、CS/AIX のサポート担当者に連絡してください。

*lua_prim_rc***LUA_INVALID_VERB**

lua_verb パラメーターも *lua_opcode* パラメーターも有効ではありませんでした。verb は実行されませんでした。

WINDOWS

*lua_prim_rc***LUA_STACK_TOO_SMALL**

アプリケーションのスタック・サイズが小さすぎて、LUA が要求を完了できません。アプリケーションのスタック・サイズを大きくしてください。

*lua_prim_rc***LUA_UNEXPECTED_DOS_ERROR**

オペレーティング・システムでエラーが発生しました。

lua_sec_rc

この値はオペレーティング・システムからの戻りコードです。ご使用のオペレーティング・システムの資料で、この戻りコードの意味を調べてください。

他の verb との関連

この verb を発行するには、SLI_BID verb が正常に完了している必要があります。

各セッションで未解決状態にできる SLI_BID は常に 1 つだけです。

SLI_BID verb が正常に完了すれば、後続の SLI_RECEIVE verb で *lua_flag1.bid_enable* パラメーターを設定することにより、SLI_BID verb を再発行することができます。この方法で verb を再発行する場合は、アプリケーション・プログラムでは、その SLI_BID verb レコードに関連付けられたストレージを解放したり変更したりしないでください。

SLI_RECEIVE と SLI_BID の両方が未解決のときにホストからメッセージが送信されると、SLI_RECEIVE は完了し、SLI_BID は進行中のままになります。

使用法と制約事項

到着する各メッセージの送信権要求は、一度だけ行われます。SLI_BID verb によって、特定のセッション・フローでデータが待機していることが通知されると、アプリケーションは SLI_RECEIVE verb を発行してそのデータを受信する必要があります。送信権要求されたメッセージが SLI_RECEIVE verb の発行によって受け入れられるまでは、後続の SLI_BID では、そのセッション・フローに到着するデータは報告されません。

複数のセッション・フローで使用可能なデータがある場合、優先順位が最も高いフロー上のデータがアプリケーションに戻されます。フローの優先順位は次のとおりです (高位から低位)。

- SSCP 急送
- LU 急送
- SSCP 通常
- LU 通常

SLI_RECEIVE verb を使って読み取られたメッセージは、着信メッセージ・キューから除去され、再びアクセスすることはできなくなります。アプリケーションは、非破壊読み取りとして SLI_BID を使用して、使用可能データのタイプを検査し、その処理方法を判断してから、後続の SLI_RECEIVE を発行してデータを収集できます。ただし、複数のフロー上でデータを受け入れるように複数の *lua_flag1* フラグを設定して SLI_RECEIVE を発行した場合、データが SLI_BID verb と SLI_RECEIVE verb 間の高優先順位フローで到着すると、SLI_BID に示されるメッセージと異なるメッセージが出されることがあります。SLI_BID に示されるメッセージと同じメッセージを必ず受信するためには、SLI_RECEIVE の *lua_flag1* フラグを、SLI_BID 応答に示されているフロー上のデータのみ受信するように設定する必要があります。

lua_data_length パラメーターは、*lua_peek_data* 内のデータの長さを示します。このパラメーターの値が 12 未満 (待ちメッセージが 12 バイトより短い) の場合は、*lua_peek_data* の残りのバイトは未定義です。アプリケーションでは、未定義のバイトを使わないようにしてください。

SLI_CLOSE

SLI_CLOSE verb は、指定された LU の LU セッションと SSCP セッションの両方を終了します。

指定パラメーター

アプリケーションが指定するパラメーターは次のとおりです。

lua_verb

LUA_VERB_SLI

lua_verb_length

LUA verb レコードの長さ (バイト数)。

これは sizeof (LUA_VERB_RECORD) に設定してください。

lua_opcode

LUA_OPCODE_SLI_CLOSE

lua_correlator

オプション。この verb をアプリケーション内の他の処理と相互に関連付けるために使用できる 4 バイトの値。LUA は、この情報を使用することも、変更することはありません。

lua_luname

セッションで使用する LU の ASCII 文字の名前。これは、SLI_OPEN verb

で戻されたアクティブな LUA セッションの LU 名 (または、未解決の SLI_OPEN verb で指定された LU 名) と一致していなければなりません。

このパラメーターは、*lua_sid* パラメーターが 0 (ゼロ) の場合にのみ必要です。*lua_sid* でセッション ID が指定されている場合は、LUA はこのパラメーターを使用しません。

このパラメーターの長さは 8 バイトにしてください。名前が 8 文字未満の場合は、右側をスペース (0x20) で埋めてください。

lua_sid セッションのセッション ID。これは、直前の SLI_OPEN verb で戻されたセッション ID と一致していなければなりません。

このパラメーターはオプションです。セッション ID を指定しない場合は、*lua_luname* パラメーターでセッションの LU 名を指定する必要があります。

lua_post_handle

AIX, LINUX

verb が非同期で完了した場合に、完了を通知するために LUA が呼び出すコールバック・ルーチンを指すポインター。

WINDOWS

SLI 関数呼び出し内で VCB が使用されている場合は、このフィールドをイベント・ハンドルに設定してください。WinSLI 関数呼び出しで VCB が使用されている場合、このフィールドは予約済みです。

lua_flag1 パラメーター

セッションを即時に閉じる場合は、*lua_flag1.close_abend* パラメーターを 1 に設定し、SLI がホストとの SNA メッセージのやりとりを正常に終了して、セッションを完全に終わってから閉じる場合は、0 (ゼロ) に設定します。正常または異常終了のクローズ処理の詳細については、129 ページの『使用法と制約事項』を参照してください。

戻りパラメーター

LUA は常に次のパラメーターを戻します。

lua_flag2.async

このフラグは、verb が非同期で完了した場合は 1 に設定され、その verb が同期して完了した場合は 0 (ゼロ) に設定されます。

その他の戻りパラメーターは、verb が正常に完了したかどうかによって異なります。以下の節を参照してください。

正常な実行

verb が正常に実行された場合、LUA は次のパラメーターを戻します。

lua_prim_rc
LUA_OK

実行の失敗

verb が正常に完了しなかった場合、LUA は、エラーのタイプを示す 1 次戻りコードと、実行が成功しなかった理由の詳細を示す 2 次戻りコードを戻します。

パラメーターの検査: 以下の戻りコードは、指定パラメーターに誤りがあったために、その *verb* が正常に完了しなかったことを示します。

lua_prim_rc
LUA_PARAMETER_CHECK

lua_sec_rc
値は次のとおりです。

LUA_BAD_SESSION_ID

lua_sid パラメーターと一致するセッション ID をもつアクティブな LUA LU セッションはありませんでした。

LUA_INVALID_LUNAME

lua_luname パラメーターで識別される LU がアクティブなノード上で見つかりませんでした。その LU 名または LU プール名が構成ファイルで定義され、それらが構成されているノードが始動されているかを確認してください。

AIX, LINUX

LUA_INVALID_POST_HANDLE

lua_post_handle パラメーターが、コールバック・ルーチンを指す有効なポインターではありませんでした。

LUA_RESERVED_FIELD_NOT_ZERO

verb レコードの予約フィールド、またはこの *verb* で使用されないパラメーターが、ゼロ以外の値に設定されていました。

LUA_VERB_LENGTH_INVALID

lua_verb_length パラメーターの値に、この *verb* に必要な *verb* レコードの長さより小さい値が指定されていました。

状態の検査: 以下の戻りコードは、*verb* が発行されたときのセッション状態では、その *verb* が無効であったことを示します。

lua_prim_rc
LUA_STATE_CHECK

lua_sec_rc
値は次のとおりです。

LUA_CLOSE_PENDING

SLI_CLOSE (正常または異常終了) がすでに進行中である場合に、アプリケーションが SLI_CLOSE (正常) を発行したか、あるいは

SLI_CLOSE

SLI_CLOSE (異常終了) がすでに進行中である場合に、SLI_CLOSE (異常終了) を発行しました。先に発行されたSLI_CLOSE (正常) に続けて発行される SLI_CLOSE (異常終了) のみが 2 番目の SLI_CLOSE として有効です。

LUA_NO_SLI_SESSION

この verb で指定された LU 名の LUA セッションがないか、あるいはセッションに障害が発生しました。

未解決の SLI_OPEN verb を取り消すために、その verb に指定された *lua_luname* パラメーターを使用して SLI_CLOSE verb が発行された場合、この戻りコードは、この verb が処理される前に SLI_OPEN が完了したことを示すことがあります。その verb が異常終了した (そのため、セッションがない) か、*lua_luname* で指定された、プールの別の LU を使って SLI_OPEN が正常に完了した (そのため、指定された LU 名のセッションがない) 可能性があります。

verb の取り消し: 以下の戻りコードは、ホストから送信されたメッセージによって取り消されたために、verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_CANCELLED

lua_sec_rc

値は次のとおりです。

LUA_RECEIVED_UNBIND_HOLD

SLI_CLOSE verb が、ホストからの UNBIND タイプ 0x02 (BIND 受信予定の UNBIND) によって取り消されました。セッションは閉じられません。アプリケーションで、SLI_BID または SLI_RECEIVE を発行して、状況情報を取得する必要があります。アプリケーションにより、SLI_OPEN verb に指定されたユーザー拡張ルーチンは、ホストが新規 BIND を送信したときに、再度呼び出されます。

LUA_RECEIVED_UNBIND_NORMAL

SLI_CLOSE verb が、ホストからの UNBIND タイプ 0x01 (正常 UNBIND) によって取り消され、セッションを開始した SLI_OPEN の *lua_session_type* パラメーターが LUA_SESSION_TYPE_DEDICATED に設定されました。セッションは閉じられません。アプリケーションで、SLI_BID または SLI_RECEIVE を発行して、状況情報を取得する必要があります。アプリケーションにより、SLI_OPEN verb に指定されたユーザー拡張ルーチンは、ホストが新規 BIND を送信したときに、再度呼び出されます。アプリケーションが、新規 BIND を待たずにセッションを終了させる場合は、SLI_CLOSE (異常終了) を発行する必要があります。

その他の状態: 以下の戻りコードは、指定された verb レコードは有効であったが、verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_UNSUCCESSFUL

lua_sec_rc

LUA_INVALID_PROCESS

この verb を発行したオペレーティング・システムのプロセスが、このセッションの SLI_OPEN verb を発行したプロセスと同じではありませんでした。セッションで verb を発行できるのは、そのセッションを開始したプロセスだけです。

LUA_NAU_INOPERATIVE

必要な SNA コンポーネント (LUA LU など) がアクティブでないか、異常終了状態です。

LUA_NO_SESSION

リモート LU との SNA セッションがアクティブではありません。

LUA_SLI_LOGIC_ERROR

この戻りコードは、次のいずれかを示します。

- ホスト・システムが SNA プロトコルに違反している
- LUA 内で内部エラーが検出された

SNA のトレース機能をアクティブにして問題を再現し (必要であればシステム管理者に連絡してください)、ホストが正しいデータを送信しているかどうかを調べてください。それでもまだ問題が解決されない場合は、CS/AIX のサポート担当者に連絡してください。

以下の戻りコードは、verb がその他の理由で正常に完了しなかったことを示します。

lua_prim_rc

LUA_COMM_SUBSYSTEM_ABENDED

CS/AIX の必須ソフトウェア構成要素 (ノードなど) が終了または停止しています。必要であれば、システム管理者に連絡してください。

lua_prim_rc

LUA_COMM_SUBSYSTEM_NOT_LOADED

Remote API Client ソフトウェアが開始されていないか、ノードが始動されていない、あるいは LUA アプリケーション向けに適切に構成されていません。CS/AIX LUA 構成パラメーターを調べて、Remote API Client とノードを始動してから、アプリケーションを実行してください。

lua_prim_rc

LUA_SESSION_FAILURE

LUA セッションに障害が発生しました。アプリケーションは、SLI_OPEN を再発行することによって、セッションを再開できません。

lua_sec_rc

値は次のとおりです。

LUA_LU_COMPONENT_DISCONNECTED

この戻りコードは、通信リンクまたはホスト LU の問題のために LUA セッションに障害が発生したことを示します。

LUA_NEGATIVE_RSP_CHASE

この戻りコードは、SLI で CHASE コマンドに対する否定応答が受信されたために、LUA セッションが閉じられたことを示します。

LUA_NEGATIVE_RSP_SHUTD

この戻りコードは、SLI で SHUTD コマンドに対する否定応答が受信されたために、LUA セッションが閉じられたことを示します。

LUA_NEGATIVE_RSP_RSHUTD

この戻りコードは、SLI で RSHUTD コマンドに対する否定応答が受信されたために、LUA セッションが閉じられたことを示します。

LUA_RECEIVED_UNBIND

この戻りコードは、UNBIND コマンドがホストから送信され、セッションが終了したことを示します。この値は、セッションに対する SLI_OPEN verb に *lua_session_type* LUA_SESSION_TYPE_DEDICATED が指定されている場合にのみ生成されます。

LUA_UNEXPECTED_SNA_SEQUENCE

この戻りコードは、SLI でホストからの予期しない SNA メッセージが受信されたために、LUA セッションが閉じられたことを示します。

lua_prim_rc

LUA_STACK_TOO_SMALL

アプリケーションのスタック・サイズが小さすぎて、LUA が要求を完了できません。アプリケーションのスタック・サイズを大きくしてください。

lua_prim_rc

LUA_UNEXPECTED_DOS_ERROR

オペレーティング・システムでエラーが発生しました。

lua_sec_rc

この値はオペレーティング・システムからの戻りコードです。ご使用のオペレーティング・システムの資料で、この戻りコードの意味を調べてください。

lua_prim_rc

LUA_INVALID_VERB

lua_verb パラメーターも *lua_opcode* パラメーターも有効ではありませんでした。verb は実行されませんでした。

他の verb との関連

この verb は、SLI_OPEN verb が発行された後であればいつでも発行できます。SLI_OPEN がまだ完了せず、アプリケーションで取り消す必要がある場合は、*lua_flag1.close_abend* を 1 (異常終了クローズを示す) に設定して SLI_CLOSE を発行して取り消しを行います。

SLI_CLOSE (正常) の保留中に、アプリケーションが、正常なクローズ処理を待たずにすぐにセッションを終了する必要があると判断した場合は、SLI_CLOSE (異常終了) を発行できます。

SLI_CLOSE が発行されたときに保留中の他の LUA verb があると、保留中の verb はそれ以上処理されず、1 次戻りコード LUA_CANCELLED を戻して完了します。

この verb が完了した後は、このセッションについて他の LUA verb を発行することはできません。アプリケーションは、同じ LU に対して、あるいは異なる LU に対して SLI_OPEN を発行して、新規セッションを開始できます。

使用法と制約事項

セッションのクローズ処理は、次に示すように、ホスト (最初に開始されたクローズ) または LUA アプリケーション (2 番目に開始されたクローズ) によって開始できます。いずれの場合も、アプリケーションは通常、`lua_flag1.close_abend` を 0 (ゼロ) に設定します。これは、LUA とホストが、通常のメッセージ・シーケンスを交換し合ってセッションを終了する正常なクローズを示します。

最初に開始されるクローズ

ホストは、SHUTD コマンドを送信することによって、クローズ処理を開始します。これは、SLI_BID verb または SLI_RECEIVE verb の `LUA_SESSION_END_REQUESTED` 状況値としてアプリケーションに戻されます。

アプリケーションでは、セッションを閉じる準備ができると、SLI_CLOSE を発行して応答します。これによって、LUA とホスト間で次のメッセージ・シーケンスが生じます。

- LUA がホストに CHASE を送信し、応答を受け取る。
- LUA がホストにシャットダウン完了 (SHUTC) を送信し、応答を受け取る。
- オプションで、ホストが CLEAR を送信すると、LUA はこれを受け取り、応答を送信する。
- ホストが UNBIND を送信し、LUA はこれを受け取り、応答を送信する。
- LUA が RUI セッションを停止し、SLI_CLOSE verb が戻される。

2 番目に開始されるクローズ

アプリケーションは、SLI_CLOSE を発行してクローズ処理を開始します。これによって、LUA とホスト間で次のメッセージ・シーケンスが生じます。

- LUA がホストに RSHUTD を送信し、応答を受け取る。
- オプションで、ホストが CLEAR を送信すると、LUA はこれを受け取り、応答を送信する。
- ホストが UNBIND を送信し、LUA はこれを受け取り、応答を送信する。
- LUA が RUI セッションを停止し、SLI_CLOSE verb が戻される。

SLI_CLOSE (正常) の進行中は、ホストから次のメッセージのいずれかを送信することによって割り込みを行うことができます。

SLI_CLOSE

- UNBIND タイプ 0x02 (BIND 受信予定の UNBIND)
- UNBIND タイプ 0x01 (正常な UNBIND)。セッションを開始した SLI_OPEN のパラメーター *lua_session_type* が `LUA_SESSION_TYPE_DEDICATED` に設定されている場合。

これらのいずれの場合も、SLI_CLOSE verb は、1 次戻りコード CANCELLED を戻します。セッションは閉じられません。アプリケーションで、SLI_BID または SLI_RECEIVE を発行して、状況情報を取得する必要があります。アプリケーションにより、SLI_OPEN verb に指定されたユーザー拡張ルーチンは、ホストが新規 BIND を送信したときに、再度呼び出されます。

アプリケーションが、通常メッセージ・シーケンスを待つことなく、すぐにセッションを終了する必要があるか、ホストが UNBIND (正常) を送信した後で、新規 BIND を待たずに専用セッションを閉じる必要がある場合は、*lua_flag1.close_abend* を 1 に設定して SLI_CLOSE を発行します。これによって、SLI セッションが終了し、LUA が必要なクリーンアップ処理をすべて行って、セッションが終了したことをホストに通知します。

lua_flag1.close_abend を 0 (ゼロ) に設定して SLI_CLOSE (正常) を発行する前に、アプリケーションでは、ホストからの未解決メッセージをすべて受信しているか、また必要な応答をすべて送信しているかを確認しなければなりません。必要な応答が送信されていない場合、LUA は、閉じるタイプを自動的に変更して、上記のように CLOSE (異常終了) 処理を実行します。

SLI_OPEN

SLI_OPEN verb は、指定された LU の SNA セッション、または指定された LU プール内で最初に使用可能な LU の SNA セッションを確立します。

指定パラメーター

アプリケーションが指定するパラメーターは次のとおりです。

lua_verb

LUA_VERB_SLI

lua_verb_length

LUA verb レコードの長さ (バイト数)。

これは sizeof (LUA_VERB_RECORD) に設定してください。

lua_opcode

LUA_OPCODE_SLI_OPEN

lua_correlator

オプション。この verb をアプリケーション内の他の処理と相互に関連付けるために使用できる 4 バイトの値。LUA は、この情報を使用することも、変更することはありません。

lua_luname

セッションを開始したい LU または LU プールの ASCII 文字の名前。こ

これは、CS/AIX に対して設定された LU タイプ 0 から 3 の LU 名、または LU プールの名前と一致していなければなりません。この名前は、次のように使用されます。

- プールに含まれていない LU の名前を指定した場合、CS/AIX はこの LU を使ってセッションの開始を試みます。アプリケーションは、各 verb ごとにそれぞれ異なる LU を指定した複数の SLI_OPEN verb を使用することにより、複数のセッションを開始できます。同じ LU で複数のセッションを開始することはできません。
- LU プールの名前を指定した場合またはプール内の LU 名を指定した場合、CS/AIX は、指定された LU が使用可能であればその LU を使ってセッションの開始を試みます。それ以外の場合は、そのプール内の最初に使用可能な LU を使ってセッションの開始を試みます。アプリケーションは、同じプールを使って複数のセッションを開始できます。CS/AIX は各セッションに、そのプール内の異なる LU を割り当てます。セッションに使用される実際の LU の名前は、SLI_OPEN verb の戻りパラメーターで示されます。

このパラメーターの長さは 8 バイトにしてください。名前が 8 文字未満の場合は、右側をスペース (0x20) で埋めてください。

lua_data_length

lua_data_ptr パラメーターに指定される不定形式ログオン・データ または INITSELF データの長さ、またはデータが提供されない場合はゼロ。

lua_data_ptr

セッションを開始するためにホストに送信されるメッセージがあれば、それへのポインター。この内容は、次のように、*lua_init_type* パラメーターによって決まります。

- *lua_init_type* が `LUA_INIT_TYPE_SEC_IS` である場合、アプリケーションは、モード名および PLU 名などの必要なユーザー情報を含む INITSELF 要求単位を提供する必要があります。
- *lua_init_type* が `LUA_INIT_TYPE_SEC_LOG` である場合、アプリケーションで、SSCP の通常フローに不定形式ログオン・メッセージを提供する必要があります。
- *lua_init_type* が `LUA_INIT_TYPE_PRIM` または `LUA_INIT_TYPE_PRIM_SSCP` である場合、このパラメーターは使用されず、アプリケーションは、ヌル・ポインターを提供する必要があります。

lua_post_handle

AIX, LINUX

verb が非同期で完了した場合に、完了を通知するために LUA が呼び出すコールバック・ルーチンを指すポインター (LUA の初期検査で verb が不合格で、SLI エントリー・ポイントからゼロが戻されると、LUA はこのルーチンを呼び出しません)。

WINDOWS

SLI 関数呼び出し内で VCB が使用されている場合は、このフィールドをイベント・ハンドルに設定してください。WinSLI 関数呼び出しで VCB が使用されている場合、このフィールドは予約済みです。



詳しくは、13 ページの『第 2 章 LUA アプリケーションの設計と作成』を参照してください。

lua_encr_decr_option

このパラメーターは予約済みであり、ゼロに設定されていなければなりません。

lua_init_type

LUA がどのようにセッションを開始する必要があるかを指定します。値は次のとおりです。

LUA_INIT_TYPE_SEC_IS

2 番目の開始: アプリケーションの INITSELF メッセージ (*lua_data_ptr* で示される) をホストに送信します。

LUA_INIT_TYPE_SEC_LOG

2 番目の開始: アプリケーションの不定形式ログオン・メッセージ (*lua_data_ptr* で示される) をホストに送信します。

LUA_INIT_TYPE_PRIM

最初の開始: ホストからの BIND を待ちます。

LUA_INIT_TYPE_PRIM_SSCP

SSCP へのアクセスによる最初の開始: アプリケーションが、SSCP 通常フローで、SLI_SEND verb および SLI_RECEIVE verb を発行して、独自の INITSELF または LOGON メッセージを提供し、それらの応答を受信することができます。アプリケーションでは、SLI_OPEN の発行後に、SLI_BID または SLI_RECEIVE を発行して、状況標識 INIT_COMPLETE を取得してから、SLI_SEND と SLI_RECEIVE を使用して、INITSELF または LOGON メッセージを送信し、それらの応答を受信できます。

lua_session_type

LUA が UNBIND タイプ X'01' (正常) を処理する方法を指定します。値は次のとおりです。

LUA_SESSION_TYPE_NORMAL

肯定応答を送信し、RUI_TERM を発行して、NOTIFY(disabled) が SSCP に送信されるようにします。SSCP-LU フローが使用不可になります。

LUA_SESSION_TYPE_DEDICATED

肯定応答を送信し、BIND コマンド、オプションで CRV および STSN コマンド、そして SDT コマンドが受信されるまで、SLI セッションを中断します。NOTIFY(disabled) は、SSCP には送信されません。この場合、アプリケーションは、ホストからの新規 BIND

を待たずに、SLI_CLOSE (異常終了) を発行して、中断されたセッションを終了することができます。

lua_wait

2 番目に開始されたセッション起動の再試行タイムアウト (秒数)。このパラメーターは、*lua_init_type* が `LUA_INIT_TYPE_PRIM` または `LUA_INIT_TYPE_PRIM_SSCP` である場合は無視されます。

最初の試行に対してホストから次のメッセージで応答が送られてくると、LUA は、このタイムアウト時間が経過してから、セッション開始を再試行します (アプリケーションの `INITSELF` または `LOGON` メッセージを再送することによって)。

- `INITSELF` または `LOGON` に対する否定応答で、2 次戻りコード `RESOURCE_NOT_AVAILABLE`、`SESSION_LIMIT_EXCEEDED`、`SSCP_LU_SESS_NOT_ACTIVE`、または `SESSION_SERVICE_PATH_ERROR` 付き。
- ネットワーク・サービス・プロシージャ・エラー (NSPE) メッセージ
- `NOTIFY` コマンド。プロシージャ・エラーを示します。

このパラメーターがゼロに設定されている場合は、LUA は、セッションの開始を再試行しません。

lua_open_extension

アプリケーションの `SLI_OPEN` 拡張ルーチンがある場合は、その情報。このパラメーターは構造体の配列で、それぞれに特定の拡張ルーチンの情報が保持されています。

アプリケーションでは、0 から 3 個の拡張ルーチンを指定できます。それらはいずれも、セッションの初期化中に、特定の `SNA` メッセージを処理するアプリケーションのルーチンを示します (*lua_routine_type* パラメーターで示されます)。これらは、配列内の連続したエレメント内に先頭から指定してください。提供するエントリーの最後のものについては、

lua_open_extension.lua_routine_type を `LUA_ROUTINE_TYPE_END` に設定して、リストの終わりであることを示す必要があります。

lua_open_extension.lua_routine_type

拡張ルーチンのタイプ。値は次のとおりです。

LUA_ROUTINE_TYPE_BIND

ホストからの `BIND` メッセージを検査し、それに応答するルーチン。

LUA_ROUTINE_TYPE_SDT

ホストからの `SDT` メッセージを検査し、それに応答するルーチン。

LUA_ROUTINE_TYPE_STSN

ホストからの `STSN` メッセージを検査し、それに応答するルーチン。

LUA_ROUTINE_TYPE_END

この値は、拡張ルーチンのリストの終わりを示します。これは、他のルーチンの直後の配列エレメント内で使用する必要があります (アプリケーションに拡張ルーチンが指定されていない場合は最初の配列エレメント内)。

AIX, LINUX

lua_open_extension.lua_routine_ptr

拡張ルーチンのエントリー・ポイントへのポインター。このパラメーターは、*lua_open_extension.lua_routine_type* が `LUA_ROUTINE_TYPE_END` に設定されている最後の配列エントリー内では使用されません。

LUA は、*lua_routine_type* パラメーターの値にしたがって、`SLI_BIND_ROUTINE verb`、`SLI_SDT_ROUTINE verb`、または `SLI_STSN_ROUTINE verb` でこのエントリー・ポイントを呼び出します。

WINDOWS

lua_open_extension.lua_module_name

拡張モジュールを含む DLL の名前。このパラメーターは、*lua_open_extension.lua_routine_type* が `LUA_ROUTINE_TYPE_END` に設定されている最後の配列エントリー内では使用されません。

lua_open_extension.lua_procedure_name

拡張モジュール DLL 内で呼び出すプロシージャ名。このパラメーターは、*lua_open_extension.lua_routine_type* が `LUA_ROUTINE_TYPE_END` に設定されている最後の配列エントリー内では使用されません。

LUA は、*lua_routine_type* パラメーターの値にしたがって、`SLI_BIND_ROUTINE verb`、`SLI_SDT_ROUTINE verb`、または `SLI_STSN_ROUTINE verb` でこのエントリー・ポイントを呼び出します。

lua_ending_delim

CS/AIX SLI インターフェースでは、このパラメーターは使用されません。これは、元は他の SLI インプリメンテーション用に作成されたアプリケーションとの互換性を保持するためのものです。

SLI エントリー・ポイントからの戻り値

`SLI_OPEN verb` は、SLI エントリー・ポイントから値が戻される唯一の `verb` です。

- この `verb` が LUA の初期検査で不合格になると (たとえば、アプリケーションが提供したパラメーターが正しくないため)、SLI 関数呼び出しは、そのことを示すために値ゼロを戻します。アプリケーションでは、*lua_prim_rc* および *lua_sec_rc* パラメーターを検査して、不合格の理由を判別する必要があります。CS/AIX では、アプリケーションが提供するコールバック・ルーチンは呼び出されません。
- 初期検査に合格すると、SLI 関数呼び出しから、新規セッションのセッション ID を表すゼロ以外の値が戻されます。*lua_init_type* が `LUA_INIT_TYPE_PRIM_SSCP`

に設定された場合、アプリケーションは、SSCP 通常フローの後続の SLI_BID verb または SLI_RECEIVE verb に対してこのセッション ID を使用でき (INIT_COMPLETE 状況標識を受信するために)、続いてこのフローの SLI_SEND verb および SLI_RECEIVE verb に対するセッション ID を使用することができます。

次いで、CS/AIX は、アプリケーションが提供するコールバック・ルーチンを他の SLI verb と同様に使用します。

戻りパラメーター

LUA は常に次のパラメーターを戻します。

lua_flag2.async

このフラグは、verb が非同期で完了した場合は 1 に設定され、その verb が同期して完了した場合は 0 (ゼロ) に設定されます。

その他の戻りパラメーターは、verb が正常に完了したかどうかによって異なります。以下の節を参照してください。

正常な実行

verb が正常に実行された場合、LUA は次のパラメーターを戻します。

lua_prim_rc

LUA_OK

lua_sid 新しいセッションのセッション ID。これは、この verb の SLI エントリー・ポイントからの戻り値と同じであり、後続の verb がこのセッションを識別するのに使用できます。

lua_luname

新しいセッションで使用される LU の名前。要求パラメーターの LU 名に LU プールが指定されていた場合、CS/AIX はこのパラメーターを使って、セッションに割り当てられた実際の LU の名前を戻します。その後の verb でそのセッションを識別するときは、(要求パラメーターで指定した名前ではなく) このパラメーターで戻された名前を使用する必要があります。

実行の失敗

verb が正常に完了しなかった場合、LUA は、エラーのタイプを示す 1 次戻りコードと、実行が成功しなかった理由の詳細を示す 2 次戻りコードを戻します。

verb の取り消し: 以下の戻りコードは、別の verb によって取り消されたためにその verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_CANCELLED

lua_sec_rc

LUA_TERMINATED

SLI_OPEN が完了する前に SLI_CLOSE verb が発行されました。

パラメーターの検査: 以下の戻りコードは、指定パラメーターに誤りがあったために、その verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_PARAMETER_CHECK

lua_sec_rc

値は次のとおりです。

LUA_DATA_LENGTH_ERROR

lua_init_type パラメーターに、2 番目に開始されたセッションが指定されましたが、アプリケーションから、ホストに送信する必要データが提供されませんでした。

LUA_INVALID_LUNAME

lua_luname パラメーターで識別される LU がアクティブなノード上で見つかりませんでした。その LU 名または LU プール名が構成ファイルで定義され、それらが構成されているノードが始動されているかを確認してください。

LUA_INVALID_OPEN_DATA

lua_init_type パラメーターが LUA_INIT_TYPE_SEC_IS に設定されましたが、*lua_data_ptr* で示されるデータ・バッファーには有効な INITSELF コマンドが含まれていませんでした。

LUA_INVALID_OPEN_INIT_TYPE

lua_init_type パラメーターが有効な値に設定されていませんでした。

LUA_INVALID_OPEN_ROUTINE_TYPE

lua_routine_type パラメーターが有効な値に設定されていませんでした。

AIX、LINUX

LUA_INVALID_POST_HANDLE

lua_post_handle パラメーターが、コールバック・ルーチンを指す有効なポインターではありませんでした。

LUA_INVALID_SESSION_TYPE

lua_session_type パラメーターが有効な値に設定されていませんでした。

LUA_INVALID_SLI_ENCR_OPTION

lua_encr_decr_option パラメーターが有効な値に設定されていませんでした。CS/AIX の場合、このパラメーターは 0 (ゼロ) に設定する必要があります。

LUA_RESERVED_FIELD_NOT_ZERO

verb レコードの予約フィールド、またはこの *verb* で使用されないパラメーターが、ゼロ以外の値に設定されていました。

LUA_VERB_LENGTH_INVALID

lua_verb_length パラメーターの値に、この *verb* に必要な *verb* レコードの長さより小さい値が指定されていました。

LUA_BAD_DATA_PTR

lua_data_ptr パラメーターに指定された値が有効ではありませんでした。

LUA_BAD_SESSION_ID

lua_sid パラメーターと一致するセッション ID をもつアクティブな LUA LU セッションはありませんでした。

状態の検査: 以下の戻りコードは、*verb* が発行されたときのセッション状態では、その *verb* が無効であったことを示します。

lua_prim_rc

LUA_STATE_CHECK

lua_sec_rc

LUA_DUPLICATE_RUI_INIT

このセッションに対して SLI_OPEN *verb* を現在処理中です。

その他の状態: 以下の戻りコードは、指定された *verb* レコードは有効であったが、*verb* が正常に完了しなかったことを示します。

lua_prim_rc

LUA_UNSUCCESSFUL

lua_sec_rc

値は次のとおりです。

LUA_COMMAND_COUNT_ERROR

LU プールの名前、またはプール内の LU の名前を *verb* で指定しましたが、そのプール内の LU はすべて使用中です。

LUA_INVALID_PROCESS

lua_luname パラメーターで指定された LU は、別のプロセスで使用されています。

LUA_LINK_NOT_STARTED

ホストへの接続が開始されていません。使用できるリンクでアクティブなものはありません。

LUA_SESSION_ALREADY_OPEN

アプリケーションが提供した名前の LU では、すでにセッションが開始しています。

LUA_NAU_INOPERATIVE

必要な SNA コンポーネント (LUA LU など) がアクティブでないか、異常終了状態です。

LUA_NO_SESSION

リモート LU との SNA セッションがアクティブではありません。

LUA_SLI_LOGIC_ERROR

この戻りコードは、次のいずれかを示します。

- ホスト・システムが SNA プロトコルに違反している
- LUA 内で内部エラーが検出された

SNA のトレース機能をアクティブにして問題を再現し (必要であればシステム管理者に連絡してください)、ホストが正しいデータを送信しているかどうかを調べてください。それでもまだ問題が解決されない場合は、CS/AIX のサポート担当者に連絡してください。

(その他の値)

その他の 2 次戻りコードはすべて、SNA センス・コードです。戻される SNA センス・コードの解釈については、38 ページの『SNA 情報』を参照してください。

次に示すセンス・コード値は CS/AIX に固有の値であり、CS/AIX 構成とホスト構成の間の不一致を示します。

0x10020000

要求された LU を所有している PU の物理装置活動化 (ACTPU) が、ホストから送信されません。

0x10110000

要求された LU の ACTLU が、ホストから送信されません。これは通常、その LU がホストで設定されていないことを示します。

0x10120000

要求された LU の ACTLU が、ホストから送信されません。ホストは従属 LU の動的定義 (DDDLU) をサポートしていますが、その LU の DDDLU 処理は失敗しました。

以下の戻りコードは、verb がその他の理由で正常に完了しなかったことを示します。

lua_prim_rc

LUA_COMM_SUBSYSTEM_NOT_LOADED

この戻りコードは、次のいずれかの状態を示します。

- Remote API Client ソフトウェアが開始されていません。アプリケーションを実行する前に、Remote API Client を開始してください。
- アクティブな CS/AIX ノードがありません。LUA verb を使用するには、要求された LU を所有しているローカル・ノードまたは要求された LU プール内の 1 つ以上の LU を所有しているローカル・ノードを始動しておく必要があります。必要であれば、システム管理者に連絡してください。

lua_prim_rc

LUA_COMM_SUBSYSTEM_ABENDED

CS/AIX の必須ソフトウェア構成要素 (ノードなど) が終了または停止しています。必要であれば、システム管理者に連絡してください。

lua_prim_rc

LUA_SESSION_FAILURE

LUA セッションに障害が発生しました。アプリケーションは、SLI_OPEN を再発行することによって、セッションを再開できません。

lua_sec_rc

値は次のとおりです。

LUA_LU_COMPONENT_DISCONNECTED

通信リンクまたはホスト LU に問題が発生したために、LUA セッションが失敗しました。

LUA_SLI_LOGIC_ERROR

この戻りコードは、次のいずれかを示します。

- ホスト・システムが SNA プロトコルに違反している
- LUA 内で内部エラーが検出された

SNA のトレース機能をアクティブにして問題を再現し (必要であればシステム管理者に連絡してください)、ホストが正しいデータを送信しているかどうかを調べてください。それでもまだ問題が解決されない場合は、CS/AIX のサポート担当者に連絡してください。

lua_prim_rc

LUA_INVALID_VERB

lua_verb パラメーターも *lua_opcode* パラメーターも有効ではありませんでした。verb は実行されませんでした。

lua_prim_rc

LUA_STACK_TOO_SMALL

アプリケーションのスタック・サイズが小さすぎて、LUA が要求を完了できません。アプリケーションのスタック・サイズを大きくしてください。

lua_prim_rc

LUA_UNEXPECTED_DOS_ERROR

オペレーティング・システムでエラーが発生しました。

lua_sec_rc

この値はオペレーティング・システムからの戻りコードです。ご使用のオペレーティング・システムの資料で、この戻りコードの意味を調べてください。

他の verb との関連

SLI_OPEN verb は、セッションで発行される最初の LUA verb でなければなりません。

この verb が正常に完了するまでは、他の LUA verb でそのセッションに対して発行できるのは、次のものだけです。

- SLI_CLOSE。 *lua_flag1.close_abend* を 1 (異常終了クローズを示す) に設定します。これは保留中の SLI_OPEN を取り消します。
- *lua_init_type* が LUA_INIT_TYPE_PRIM_SSCP に設定された場合:
 - SLI_BID または SLI_RECEIVE。 INIT_COMPLETE 状況標識を取得します。
 - SSCP 通常フロー・データに対する SLI_SEND および SLI_RECEIVE。 INITSELF または LOGON メッセージを送信し、それらの応答を受信します。

SLI_OPEN

そのセッションで発行するその他のすべての verb では、この verb で戻される次のパラメーターのいずれかを使ってセッションを指定する必要があります。

- セッション ID。これは、*lua_sid* パラメーターで (また、SLI エントリー・ポイントからの戻り値として) アプリケーションに戻されます。
- LU 名。これは、*lua_luname* パラメーターでアプリケーションに戻されます。

使用法と制約事項

SLI_OPEN verb が正常に完了すると、そのセッションではセッション開始時に指定された LU が使用されます。SLI_CLOSE verb が発行されるか、LUA_SESSION_FAILURE 1 次戻りコードが戻されるまでは、(このアプリケーションまたは他のアプリケーションの) その他の LUA セッションではその LU を使用することはできません。

SLI_OPEN verb が 1 次戻りコード LUA_IN_PROGRESS で戻った場合は、セッション ID が *lua_sid* パラメーターで戻されます。このセッション ID は、verb が正常に完了した場合に戻されるものと同じで、セッションで別の verb を発行するために使用できます。

SLI_PURGE

SLI_PURGE verb は、直前の SLI_RECEIVE を取り消します。SLI_RECEIVE は、*lua_flag1.nowait* (即時に戻る) オプションを指定せずに送信された場合、指定されたフローに使用可能なデータがないと、無期限に待機します。SLI_PURGE は、この待機中の verb を強制的に戻します (1 次戻りコード LUA_CANCELLED を指定)。

指定パラメーター

アプリケーションが指定するパラメーターは次のとおりです。

lua_verb

LUA_VERB_SLI

lua_verb_length

LUA verb レコードの長さ (バイト数)。

これは sizeof (LUA_VERB_RECORD) に設定してください。

lua_opcode

LUA_OPCODE_SLI_PURGE

lua_correlator

オプション。この verb をアプリケーション内の他の処理と相互に関連付けるために使用できる 4 バイトの値。LUA は、この情報を使用することも、変更することはありません。

lua_luname

セッションで使用する LU の ASCII 文字の名前。これは、SLI_OPEN verb で戻された、アクティブな LUA セッションの LU 名と一致していなければなりません。

このパラメーターは、*lua_sid* パラメーターが 0 (ゼロ) の場合にのみ必要です。*lua_sid* でセッション ID が指定されている場合は、LUA はこのパラメーターを使用しません。

このパラメーターの長さは 8 バイトにしてください。名前が 8 文字未満の場合は、右側をスペース (0x20) で埋めてください。

lua_sid セッションのセッション ID。これは、直前の SLI_OPEN verb で戻されたセッション ID と一致していなければなりません。

このパラメーターはオプションです。セッション ID を指定しない場合は、*lua_luname* パラメーターでセッションの LU 名を指定する必要があります。

lua_data_ptr

除去する SLI_RECEIVE VCB を指すポインター。

lua_post_handle

AIX, LINUX

verb が非同期で完了した場合に、完了を通知するために LUA が呼び出すコールバック・ルーチンを指すポインター。

WINDOWS

SLI 関数呼び出し内で VCB が使用されている場合は、このフィールドをイベント・ハンドルに設定してください。WinSLI 関数呼び出しで VCB が使用されている場合、このフィールドは予約済みです。

詳しくは、13 ページの『第 2 章 LUA アプリケーションの設計と作成』を参照してください。

戻りパラメーター

LUA は常に次のパラメーターを戻します。

lua_flag2.async

このフラグは、verb が非同期で完了した場合は 1 に設定され、その verb が同期して完了した場合は 0 (ゼロ) に設定されます。

その他の戻りパラメーターは、verb が正常に完了したかどうかによって異なります。以下の節を参照してください。

正常な実行

verb が正常に完了した場合は、次のパラメーターが戻されます。

lua_prim_rc

LUA_OK

lua_sid この verb の発行時に、アプリケーションがセッション ID ではなく

lua_luname パラメーターを指定した場合は、LUA はセッション ID を戻します。

実行の失敗

`verb` が正常に完了しなかった場合、LUA は、エラーのタイプを示す 1 次戻りコードと、実行が成功しなかった理由の詳細を示す 2 次戻りコードを戻します。

verb の取り消し: 以下の戻りコードは、別の `verb` によって取り消されたためにその `verb` が正常に完了しなかったことを示します。

lua_prim_rc
LUA_CANCELLED

lua_sec_rc
LUA_TERMINATED
この `verb` の保留中に SLI_CLOSE `verb` が発行されました。

パラメーターの検査: 以下の戻りコードは、指定パラメーターに誤りがあったために、その `verb` が正常に完了しなかったことを示します。

lua_prim_rc
LUA_PARAMETER_CHECK

lua_sec_rc
値は次のとおりです。

LUA_BAD_DATA_PTR
lua_data_ptr パラメーターが 0 (ゼロ) に設定されていました。

LUA_BAD_SESSION_ID
lua_sid パラメーターと一致するセッション ID をもつアクティブな LUA LU セッションはありませんでした。

AIX, LINUX

LUA_INVALID_POST_HANDLE
lua_post_handle パラメーターが、コールバック・ルーチンを指す有効なポインターではありませんでした。

LUA_RESERVED_FIELD_NOT_ZERO
`verb` レコードの予約フィールド、またはこの `verb` で使用されないパラメーターが、ゼロ以外の値に設定されていました。

LUA_VERB_LENGTH_INVALID
lua_verb_length パラメーターの値に、この `verb` に必要な `verb` レコードの長さより小さい値が指定されていました。

状態の検査: 以下の戻りコードは、`verb` が発行されたときのセッション状態では、その `verb` が無効であったことを示します。

lua_prim_rc
LUA_STATE_CHECK

lua_sec_rc
値は次のとおりです。

LUA_NO_RECEIVE_TO_PURGE

lua_data_ptr パラメーターが前の SLI_RECEIVE VCB のアドレスに設定されていませんでした。

LUA_NO_SLI_SESSION

この verb で指定された LU 名では SLI_OPEN verb がまだ正常に完了していないか、あるいはセッションに障害が発生しました。

LUA_SLI_PURGE_PENDING

この verb が発行されたときに、SLI_PURGE verb がすでに保留中でした。同時に未解決にしておける SLI_PURGE は 1 つだけです。

その他の状態: 以下の戻りコードは、指定された verb レコードは有効であったが、verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_UNSUCCESSFUL

lua_sec_rc

値は次のとおりです。

LUA_INVALID_PROCESS

この verb を発行したオペレーティング・システムのプロセスが、このセッションの SLI_OPEN verb を発行したプロセスと同じではありませんでした。セッションで verb を発行できるのは、そのセッションを開始したプロセスだけです。

LUA_NO_RECEIVE_TO_PURGE

アプリケーションが SLI_PURGE を発行する前に、直前の SLI_RECEIVE verb が完了しました。これはエラー状態ではないため、エラー報告をしないでこの状態を処理するようにアプリケーションを設計する必要があります。

LUA_NAU_INOPERATIVE

必要な SNA コンポーネント (LUA LU など) がアクティブでないか、異常終了状態です。

LUA_NO_SESSION

リモート LU との SNA セッションがアクティブではありません。

LUA_SLI_LOGIC_ERROR

この戻りコードは、次のいずれかを示します。

- ホスト・システムが SNA プロトコルに違反している
- LUA 内で内部エラーが検出された

SNA のトレース機能をアクティブにして問題を再現し (必要であればシステム管理者に連絡してください)、ホストが正しいデータを送信しているかどうかを調べてください。それでもまだ問題が解決されない場合は、CS/AIX のサポート担当者に連絡してください。

以下の戻りコードは、verb がその他の理由で正常に完了しなかったことを示します。

lua_prim_rc

LUA_COMM_SUBSYSTEM_ABENDED

CS/AIX の必須ソフトウェア構成要素 (ノードなど) が終了または停止しています。必要であれば、システム管理者に連絡してください。

lua_prim_rc

LUA_COMM_SUBSYSTEM_NOT_LOADED

Remote API Client ソフトウェアが開始されていないか、ノードが始動されていない、あるいは LUA アプリケーション向けに適切に構成されていません。CS/AIX LUA 構成パラメーターを調べて、Remote API Client とノードを始動してから、アプリケーションを実行してください。

lua_prim_rc

LUA_SESSION_FAILURE

LUA セッションに障害が発生しました。アプリケーションは、SLI_OPEN を再発行することによって、セッションを再開できません。

lua_sec_rc

値は次のとおりです。

LUA_LU_COMPONENT_DISCONNECTED

この戻りコードは、通信リンクまたはホスト LU の問題のために LUA セッションに障害が発生したことを示します。

LUA_RECEIVED_UNBIND

この戻りコードは、UNBIND コマンドがホストから送信され、セッションが終了したことを示します。この値は、セッションに対する SLI_OPEN verb に *lua_session_type* LUA_SESSION_TYPE_DEDICATED が指定されている場合にのみ生成されます。

lua_prim_rc

LUA_INVALID_VERB

lua_verb パラメーターも *lua_opcode* パラメーターも有効ではありませんでした。verb は実行されませんでした。

lua_prim_rc

LUA_STACK_TOO_SMALL

アプリケーションのスタック・サイズが小さすぎて、LUA が要求を完了できません。アプリケーションのスタック・サイズを大きくしてください。

lua_prim_rc

LUA_UNEXPECTED_DOS_ERROR

オペレーティング・システムでエラーが発生しました。

lua_sec_rc

この値はオペレーティング・システムからの戻りコードです。ご使用のオペレーティング・システムの資料で、この戻りコードの意味を調べてください。

他の verb との関連

この verb は、SLI_RECEIVE がすでに発行されていて、完了が保留されている (すなわち、1 次戻りコードが IN_PROGRESS である) 場合にだけ使用できます。

SLI_RECEIVE

SLI_RECEIVE verb は、ホストからアプリケーション LU に送信されるデータの完全なチェーンまたは状況情報を受信します。

データの読み取り元として特定の 1 つのメッセージ・フロー (LU 通常、LU 急送、SSCP 通常、または SSCP 急送) を指定するか、あるいは複数のメッセージ・フローを指定することができます。2 つの SLI_RECEIVE verb が同じフローを指定していないかぎり、複数の SLI_RECEIVE verb が未解決であってもかまいません。

指定パラメーター

アプリケーションが指定するパラメーターは次のとおりです。

lua_verb

LUA_VERB_SLI

lua_verb_length

LUA verb レコードの長さ (バイト数)。

これは sizeof (LUA_VERB_RECORD) に設定してください。

lua_opcode

LUA_OPCODE_SLI_RECEIVE

lua_correlator

オプション。この verb をアプリケーション内の他の処理と相互に関連付けるために使用できる 4 バイトの値。LUA は、この情報を使用することも、変更することはありません。

lua_luname

セッションで使用する LU の ASCII 文字の名前。これは、SLI_OPEN verb で戻された、アクティブな LUA セッションの LU 名と一致していなければなりません。

このパラメーターは、*lua_sid* パラメーターが 0 (ゼロ) の場合にのみ必要です。*lua_sid* でセッション ID が指定されている場合は、LUA はこのパラメーターを使用しません。

このパラメーターの長さは 8 バイトにしてください。名前が 8 文字未満の場合は、右側をスペース (0x20) で埋めてください。

lua_sid セッションのセッション ID。これは、直前の SLI_OPEN verb で戻されたセッション ID と一致していなければなりません。

このパラメーターはオプションです。セッション ID を指定しない場合は、*lua_luname* パラメーターでセッションの LU 名を指定する必要があります。

lua_max_length

データを受信するために提供されるバッファの長さ。

lua_data_ptr

データを受信するために提供されるバッファを指すポインター。

lua_post_handle

AIX, LINUX

verb が非同期で完了した場合に、完了を通知するために LUA が呼び出すコールバック・ルーチンを指すポインター。

WINDOWS

SLI 関数呼び出し内で VCB が使用されている場合は、このフィールドをイベント・ハンドルに設定してください。WinSLI 関数呼び出しで VCB が使用されている場合、このフィールドは予約済みです。

詳しくは、13 ページの『第 2 章 LUA アプリケーションの設計と作成』を参照してください。

lua_flag1 パラメーター

読み取り可能なデータであるかどうかに関係なく SLI_RECEIVE *verb* をできるだけ早く戻したい場合は、*lua_flag1.nowait* パラメーターを 1 に設定してください。データが使用可能になるのを待ってから *verb* を戻したい場合は、0 (ゼロ) に設定してください。

注:

1. *lua_flag1.nowait* パラメーターに 1 を設定するということは、*verb* が同期完了するというものではありません。LUA ライブラリーは、ローカル・ノードと通信して、データが使用可能かそうでないかを判別する必要があります。そのためには、アプリケーションのロックを妨ぐために、*verb* は非同期で戻らなければなりません。このパラメーターは、すぐに使用できるデータがない場合、*verb* はそのことを示すためにできるだけ早く非同期で戻る、ということの意味しています。
2. アプリケーションが SLI_RECEIVE を発行したときに、複数 RU チェーンの最初の RU が使用可能であると、*lua_flag1.nowait* パラメーターは無視されます。SLI_RECEIVE は、データのチェーンが完全に到着するまで待ってから、戻ります。

最後に実行した SLI_BID *verb* を再度使用可能にする場合は、*lua_flag1.bid_enable* パラメーターを 1 に設定してください (これは、前と全く同じパラメーターを指定して SLI_BID をもう一度発行するのと同じです)。SLI_BID を再度使用可能にしない場合は、0 (ゼロ) に設定してください。直前の SLI_BID を再度使用可能にすると、もともとその *verb* に割り振られていた VCB が再利用されます。したがって、その VCB を解放したり、変更したりしないでください (詳しくは、154 ページの『他の *verb* との関連』を参照してください)。

次のフラグのうちの 1 つ以上を 1 に設定して、どのメッセージ・フローからデータを読み取るかを指定してください。

lua_flag1.sscp_exp

lua_flag1.lu_exp

lua_flag1.sscp_norm

lua_flag1.lu_norm

複数のフラグを設定した場合、使用可能なデータのうちに最も優先順位の高いデータが戻されます。優先順位は、(高い順から) SSCP 急送、LU 急送、SSCP 通常、LU 通常です。*lua_flag2* グループの対応するフラグが設定されて、どのフローからデータが読み取られたかが示されます (『戻りパラメーター』を参照)。

CS/AIX にインプリメントされた LUA では、SSCP 急送フローのデータは戻されません。アプリケーションでは、他のシステムにインプリメントされた LUA との互換性を保つために *sscp_exp* フラグを設定できますが、このフローではデータは戻されません。

戻りパラメーター

LUA は、常に次のパラメーターを戻します。

lua_flag2.async

このフラグは、*verb* が非同期で完了した場合は 1 に設定され、その *verb* が同期して完了した場合は 0 (ゼロ) に設定されます。

lua_flag2.bid_enable

このパラメーターは、SLI_BID が正常に再度使用可能になった場合は 1 に設定され、SLI_BID が再度使用可能にならなかった場合は 0 に設定されます。

その他の戻りパラメーターは、*verb* が正常に完了したかどうかによって異なります。以下の節を参照してください。

正常な実行またはデータの切り捨て

verb が正常に実行された場合、LUA は次のパラメーターを戻します。

lua_prim_rc

LUA_OK

verb が正常に完了した場合は、次のパラメーターが戻されます。また、指定された *lua_data_length* パラメーターの値が小さすぎたために、データが切り捨てられて *verb* が戻る場合も、これらのパラメーターが戻されます (152 ページの『その他の状態』を参照)。

lua_sid この *verb* の発行時に、アプリケーションがセッション ID ではなく

lua_luname パラメーターを指定した場合は、LUA はセッション ID を戻します。

lua_data_length

受け取ったデータの長さ。LUA は、*lua_data_ptr* で指定されたバッファーにデータを格納します。

SLI_RECEIVE

lua_rh.rrl がオフ (要求単位)で、*lua_rh.sdi* がオン (センス・データが含まれる) である場合は、LUA が、ホストから送信される要求単位を例外要求 (EXR) に変換したことを示します。この場合、データ・バッファのバイト 0 から 3 には例外に関連したセンス・データが含まれ、バイト 4 から 6 には、元の要求単位の先頭 3 バイトまでが含まれます。

lua_th 受信メッセージの伝送ヘッダー (TH) の情報。

lua_rh 受信メッセージの要求応答ヘッダー (RH) の情報。

lua_message_type

受信メッセージのメッセージ・タイプ。これは次のいずれかです。

LUA_MESSAGE_TYPE_LU_DATA
LUA_MESSAGE_TYPE_SSCP_DATA
LUA_MESSAGE_TYPE_RSP
LUA_MESSAGE_TYPE_BID
LUA_MESSAGE_TYPE_BIS
LUA_MESSAGE_TYPE_CANCEL
LUA_MESSAGE_TYPE_CHASE
LUA_MESSAGE_TYPE_LUSTAT_LU
LUA_MESSAGE_TYPE_LUSTAT_SSCP
LUA_MESSAGE_TYPE_QC
LUA_MESSAGE_TYPE_QEC
LUA_MESSAGE_TYPE_RELQ
LUA_MESSAGE_TYPE_RTR
LUA_MESSAGE_TYPE_SBI
LUA_MESSAGE_TYPE_SIGNAL

lua_flag2 パラメーター

次のフラグのどれかが 1 に設定され、データがどのメッセージ・フローで受信されたかを示します。

lua_flag2.lu_exp

lua_flag2.sscp_norm

lua_flag2.lu_norm

CS/AIX にインプリメントされた LUA では、SSCP 急送フローでデータが戻されることはありません。したがって、*sscp_exp* フラグが設定されることはありません (他のシステムにインプリメントされた LUA によって設定される場合もあります)。

正常な実行: 状況情報

注: SLI_RECEIVE が状況情報を戻すことができるのは、未解決の SLI_BID verb がない場合のみです。状況情報が使用可能なときに、両方の verb が進行中であると、状況は SLI_BID verb で戻され、SLI_RECEIVE の進行は続けられます。

verb により、データでなく、LUA 状況情報が戻された場合、LUA は次のパラメータを戻します。

lua_prim_rc
LUA_STATUS

lua_sec_rc

LUA_READY

SLI セッションは、追加のコマンドを処理する準備ができています。この状況情報が使用されるのは、直前の LUA_NOT_READY 状況が報告された後か、あるいは、*lua_prim_rc* を LUA_CANCELLED に、*lua_sec_rc* を RECEIVED_UNBIND_HOLD または RECEIVED_UNBIND_NORMAL に設定して、SLI_CLOSE verb が完了した後です。

LUA_NOT_READY

SLI セッションが、次のいずれかの理由で一時的に中断状態になっています。

- CLEAR コマンドを受信した。SDT コマンドを受信すると、セッションは再開します。
- タイプ X'02' (BIND 受信予定) の UNBIND コマンドを受信した。セッションは、BIND、オプションの CRV および STSN、そして SDT などのコマンドを受信するまで中断状態になり、SDT を受信すると再開します。元の SLI_OPEN verb が提供したどのユーザー拡張ルーチンも再度呼び出されます。
- タイプ X'01' (通常) の UNBIND コマンドを受信し、このセッションの SLI_OPEN verb に *lua_session_type* LUA_SESSION_TYPE_DEDICATED が指定されていた。セッションは、BIND、オプションの CRV および STSN、そして SDT などのコマンドを受信するまで中断状態になり、SDT を受信すると再開します。元の SLI_OPEN verb が提供したどのユーザー拡張ルーチンも再度呼び出されます。

セッションの再開時に、アプリケーションは、他の SLI_BID または SLI_RECEIVE を発行して、READY 状況を受け取る必要があります。セッション状況が LUA_NOT_READY であっても、SLI_SEND verb と SLI_RECEIVE verb を SSCP 通常フローのデータに対して発行し続けることができます。

LUA_INIT_COMPLETE

アプリケーションは、LUA_OPEN_TYPE_PRIM_SSCP タイプを指定して SLI_OPEN を発行し、すでに発行されていた RUI_INIT verb は完了しています。アプリケーションは、SSCP 通常フローのデータに対して SLI_SEND verb および SLI_RECEIVE verb を発行できるようになりました。

LUA_SESSION_END_REQUESTED

ホストがアプリケーションに対して、セッションのシャットダウンを要求する SHUTD コマンドを送信しました。アプリケーションは、セッションを閉じる準備ができ次第、SLI_CLOSE を発行する必要があります。

lua_sid この verb の発行時に、アプリケーションがセッション ID ではなく
lua_luname パラメーターを指定した場合は、LUA はセッション ID を戻します。

実行の失敗

verb が正常に完了しなかった場合、LUA は、エラーのタイプを示す 1 次戻りコードと、実行が成功しなかった理由の詳細を示す 2 次戻りコードを戻します。

verb の取り消し: 以下の戻りコードは、他の verb またはホストから送信されたメッセージによって取り消されたために、その verb が正常に完了しなかったことを示します。

lua_prim_rc
 LUA_CANCELLED

lua_sec_rc
 値は次のとおりです。

LUA_PURGED
 SLI_RECEIVE verb は、SLI_PURGE verb によって取り消されています。

LUA_TERMINATED
 この verb の保留中に SLI_CLOSE verb が発行されました。

LUA_CANCEL_COMMAND_RECEIVED
 ホストは、CANCEL コマンドを送信して、受信中のデータ・チェー
 ンの残りを取り消します。

パラメーターの検査: 以下の戻りコードは、指定パラメーターに誤りがあったため
 に、その verb が正常に完了しなかったことを示します。

lua_prim_rc
 LUA_PARAMETER_CHECK

lua_sec_rc
 値は次のとおりです。

LUA_BAD_DATA_PTR
lua_data_ptr パラメーターに指定された値が有効ではありませんで
 した。

LUA_BAD_SESSION_ID
lua_sid パラメーターと一致するセッション ID をもつアクティブな
 LUA LU セッションはありませんでした。

LUA_BID_ALREADY_ENABLED
 SLI_BID verb を再度使用可能にするよう、*lua_flag1.bid_enable* パラ
 メーターが設定されましたが、直前の SLI_BID verb がまだ進行中
 でした。

LUA_INVALID_FLOW
lua_flag1 フロー・フラグが何も設定されていませんでした。これら
 のフラグのうち 1 つ以上を 1 に設定して、どのフローから読み取る
 かを指定する必要があります。

AIX、LINUX

LUA_INVALID_POST_HANDLE

lua_post_handle パラメーターが、コールバック・ルーチンを指す有効なポインターではありませんでした。

LUA_NO_PREVIOUS_BID_ENABLED

SLI_BID verb を再度使用可能にするよう、*lua_flag1.bid_enable* パラメーターが設定されましたが、使用可能にできる直前の SLI_BID verb はありませんでした (詳しくは、154 ページの『他の verb との関連』を参照してください)。

LUA_RESERVED_FIELD_NOT_ZERO

verb レコードの予約フィールド、またはこの verb で使用されないパラメーターが、ゼロ以外の値に設定されていました。

LUA_VERB_LENGTH_INVALID

lua_verb_length パラメーターの値に、この verb に必要な verb レコードの長さより小さい値が指定されていました。

状態の検査: 以下の戻りコードは、verb が発行されたときのセッション状態では、その verb が無効であったことを示します。

lua_prim_rc
LUA_STATE_CHECK

lua_sec_rc
値は次のとおりです。

LUA_NO_SLI_SESSION

この verb で指定された LU 名では SLI_OPEN verb がまだ正常に完了していないか、あるいはセッションに障害が発生しました。

LUA_RECEIVE_ON_FLOW_PENDING

lua_flag1 グループのフロー・フラグで、未解決状態の SLI_RECEIVE verb がすでにある 1 つ以上のセッション・フローが指定されました。各セッション・フローで同時に待ち状態にしておける SLI_RECEIVE は 1 つだけです。

ホストへの否定応答の送信: 次の 1 次戻りコードは、CS/AIX で、ホストから受け取ったデータにエラーが検出されたことを示します。CS/AIX は、受信メッセージを SLI_RECEIVE verb でアプリケーションに渡す代わりに、そのメッセージを廃棄し、ホストに否定応答を送信します。LUA は、その後の SLI_RECEIVE verb または SLI_BID verb で、否定応答が送信されたことをアプリケーションに通知します。

lua_prim_rc
LUA_NEGATIVE_RSP

lua_sec_rc
否定応答でホストに送信されたセンス・コード。これは、CS/AIX がホス

ト・データのエラーを検出し、ホストに否定応答を送信したことを示します。戻されるセンス・コードの解釈については、38ページの『SNA 情報』を参照してください。

その他の状態: 以下の戻りコードは、指定された verb レコードは有効であったが、verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_UNSUCCESSFUL

lua_sec_rc

値は次のとおりです。

LUA_DATA_TRUNCATED

lua_data_length パラメーターに、メッセージで受信したデータの実際の長さより小さい値が指定されていました。*lua_data_length* で指定されたバイト数のデータだけが verb に対して戻され、残りのデータは廃棄されました。この 2 次戻りコードが戻された場合は、その他にもパラメーターが戻されます。147ページの『正常な実行またはデータの切り捨て』を参照してください。

LUA_NO_DATA

lua_flag1.nowait パラメーターが設定され、データを待たずにただちに制御を戻すように指示されました。指定されたセッション・フローには、その時点で使用可能なデータはありませんでした。

LUA_INVALID_PROCESS

この verb を発行したオペレーティング・システムのプロセスが、このセッションの SLI_OPEN verb を発行したプロセスと同じではありませんでした。セッションで verb を発行できるのは、そのセッションを開始したプロセスだけです。

LUA_NAU_INOPERATIVE

必要な SNA コンポーネント (LUA LU など) がアクティブでないか、異常終了状態です。

LUA_NO_SESSION

リモート LU との SNA セッションがアクティブではありません。

LUA_SLI_LOGIC_ERROR

この戻りコードは、次のいずれかを示します。

- ホスト・システムが SNA プロトコルに違反している
- LUA 内で内部エラーが検出された

SNA のトレース機能をアクティブにして問題を再現し (必要であればシステム管理者に連絡してください)、ホストが正しいデータを送信しているかどうかを調べてください。それでもまだ問題が解決されない場合は、CS/AIX のサポート担当者に連絡してください。

以下の戻りコードは、verb がその他の理由で正常に完了しなかったことを示します。

lua_prim_rc

LUA_COMM_SUBSYSTEM_ABENDED

CS/AIX の必須ソフトウェア構成要素 (ノードなど) が終了または停止しています。必要であれば、システム管理者に連絡してください。

lua_prim_rc

LUA_COMM_SUBSYSTEM_NOT_LOADED

Remote API Client ソフトウェアが開始されていないか、ノードが始動されていない、あるいは LUA アプリケーション向けに適切に構成されていません。CS/AIX LUA 構成パラメーターを調べて、Remote API Client とノードを始動してから、アプリケーションを実行してください。

lua_prim_rc

LUA_SESSION_FAILURE

LUA セッションに障害が発生しました。アプリケーションは、SLI_OPEN を再発行することによって、セッションを再開できません。

lua_sec_rc

値は次のとおりです。

LUA_LU_COMPONENT_DISCONNECTED

この戻りコードは、通信リンクまたはホスト LU の問題のために LUA セッションに障害が発生したことを示します。

LUA_RECEIVED_UNBIND

この戻りコードは、UNBIND コマンドがホストから送信され、セッションが終了したことを示します。この値は、セッションに対する SLI_OPEN verb に *lua_session_type* LUA_SESSION_TYPE_DEDICATED が指定されている場合にのみ生成されます。

LUA_RUI_WRITE_FAILURE

この SLI verb の処理中に使用される RUI_WRITE verb が失敗し、予期しないエラー戻りコードが生成されました。

lua_prim_rc

LUA_INVALID_VERB

lua_verb パラメーターも *lua_opcode* パラメーターも有効ではありませんでした。verb は実行されませんでした。

lua_prim_rc

LUA_STACK_TOO_SMALL

アプリケーションのスタック・サイズが小さすぎて、LUA が要求を完了できません。アプリケーションのスタック・サイズを大きくしてください。

lua_prim_rc

LUA_UNEXPECTED_DOS_ERROR

オペレーティング・システムでエラーが発生しました。

lua_sec_rc

この値はオペレーティング・システムからの戻りコードです。ご使用のオペレーティング・システムの資料で、この戻りコードの意味を調べてください。

他の verb との関連

この verb を発行するには、SLI_OPEN verb が正常に完了している必要があります。

既存の SLI_RECEIVE が保留状態であれば、保留中の SLI_RECEIVE とは別の (単数または複数の) セッション・フローを指定する場合にのみ SLI_RECEIVE をもう 1 つ発行することができます。同じセッション・フローに対して複数の SLI_RECEIVE を未解決にしておくことはできません。

lua_flag1.bid_enable パラメーターは、次の条件が満たされている場合にのみ使用できます。

- SLI_BID がすでに正常に発行され、完了している
- SLI_BID verb に割り振られたストレージがまだ解放も変更もされていない
- 保留中の SLI_BID が他にない

このパラメーターを使用して直前の SLI_BID を再度使用可能にする場合、SLI_RECEIVE のメッセージ・フロー・フラグのうち少なくとも 1 つを設定して、アプリケーションがどのフロー (1 つ以上) でデータを受け取るかを指定する必要があります。最初に受信されるデータが、SLI_RECEIVE verb によって受け入れられるフロー上にある場合、SLI_RECEIVE はそのデータと共に戻りますが、SLI_BID は戻りません。それ以外の場合は、SLI_BID は戻り、読み取り対象のデータがあることを示します (SLI_BID はすべてのフロー上のデータを受け入れるため、SLI_RECEIVE がデータを受け入れない場合は必ず SLI_BID がデータを受け入れます)。この場合、アプリケーションは該当するフローに対して別の SLI_RECEIVE を発行し、データを取得する必要があります。

特定のフロー上のデータを SLI_BID に優先して SLI_RECEIVE で処理するのではなく、すべてのフロー上のデータを SLI_BID で処理する場合は、SLI_RECEIVE を使用する代わりに、SLI_BID を明示的に再発行して、直前の SLI_BID を再度使用可能にします。

使用法と制約事項

受け取ったデータが *lua_max_length* パラメーターで指定された長さより長い場合は、データは切り捨てられ、*lua_max_length* で指定されたバイト数のデータだけが戻されます。1 次戻りコード LUA_UNSUCCESSFUL および 2 次戻りコード LUA_DATA_TRUNCATED も戻されます。

SLI_RECEIVE verb が、*lua_flag1* 内にビットを設定して、複数フローからのデータを受け入れるようにし、指定された複数のフローに使用可能なデータがある場合は、最も優先順位の高いフローのデータがアプリケーションに戻されます。フローの優先順位は次のとおりです (高位から低位)。

- SSCP 急送
- LU 急送

- SSCP 通常
- LU 通常

SLI_RECEIVE verb を使って読み取られたメッセージは、着信メッセージ・キューから除去され、再びアクセスすることはできなくなります。アプリケーションは、非破壊読み取りとして SLI_BID を使用して、使用可能データのタイプを検査し、その処理方法を判断してから、後続の SLI_RECEIVE を発行してデータを収集できます。ただし、複数のフロー上でデータを受け入れるように複数の *lua_flag1* フラグを設定して SLI_RECEIVE を発行した場合、データが SLI_BID verb と SLI_RECEIVE verb 間の高優先順位フローで到着すると、SLI_BID に示されるメッセージと異なるメッセージが出されることがあります。SLI_BID に示されるメッセージと同じメッセージを必ず受信するためには、SLI_RECEIVE の *lua_flag1* フラグを、SLI_BID 応答に示されているフロー上のデータのみ受信するように設定する必要があります。

LUA アプリケーションがメッセージであふれないようにするために、1 次と 2 次間のハーフ・セッション（これはホスト構成で指定されます）でペーシングを使用できます。LUA アプリケーションがメッセージを読み取る速度が遅い場合、CS/AIX はホストに対するペーシング応答の送信を遅らせて、ホスト側の速度を遅くします。

SLI_SEND

SLI_SEND verb は、LU セッションまたは SSCP セッションを通して、LUA アプリケーションからホストへ SNA 要求単位または応答単位を送信します。

アプリケーションでは、同時に未解決にしておける SLI_SEND verb は最大 2 つまでですが、これらは異なるセッション・フロー内になければなりません。

指定パラメーター

アプリケーションが指定するパラメーターは次のとおりです。

lua_verb

LUA_VERB_SLI

lua_verb_length

LUA verb レコードの長さ (バイト数)。

これは sizeof (LUA_VERB_RECORD) に設定してください。

lua_opcode

LUA_OPCODE_SLI_SEND

lua_correlator

オプション。この verb をアプリケーション内の他の処理と相互に関連付けるために使用できる 4 バイトの値。LUA は、この情報を使用することも、変更することはありません。

lua_luname

セッションで使用する LU の ASCII 文字の名前。これは、SLI_OPEN verb で戻された、アクティブな LUA セッションの LU 名と一致していなければなりません。

このパラメーターは、*lua_sid* パラメーターが 0 (ゼロ) の場合にのみ必要です。*lua_sid* でセッション ID が指定されている場合は、LUA はこのパラメーターを使用しません。

このパラメーターの長さは 8 バイトにしてください。名前が 8 文字未満の場合は、右側をスペース (0x20) で埋めてください。

lua_sid セッションのセッション ID。これは、直前の SLI_OPEN verb で戻されたセッション ID と一致していなければなりません。

このパラメーターはオプションです。セッション ID を指定しない場合は、*lua_luname* パラメーターでセッションの LU 名を指定する必要があります。

lua_data_length

提供されるデータの長さ。

肯定応答を送信する場合は、通常、このパラメーターは 0 (ゼロ) に設定します。LUA は、指定された順序番号に基づいて応答を完了します。BIND または STSN に対する肯定応答の場合は拡張応答を使用できるので、ゼロ以外の値を使用できます。

否定応答を送信する場合は、このパラメーターはデータ・バッファーに提供される SNA センス・コードの長さ (4 バイト) に設定してください。

lua_data_ptr

提供されるデータを格納するバッファーを指すポインター。

要求、またはデータを必要とする肯定応答の場合、バッファーには RU 全体が格納されます。RU の長さは *lua_data_length* で指定してください。

否定応答の場合は、バッファーには SNA センス・コードが格納されます。

lua_post_handle

AIX, LINUX

verb が非同期で完了した場合に、完了を通知するために LUA が呼び出すコールバック・ルーチンを指すポインター。

WINDOWS

SLI 関数呼び出し内で VCB が使用されている場合は、このフィールドをイベント・ハンドルに設定してください。WinSLI 関数呼び出しで VCB が使用されている場合、このフィールドは予約済みです。

詳しくは、13 ページの『第 2 章 LUA アプリケーションの設計と作成』を参照してください。

lua_th.snf

これは、応答を送信する場合にのみ必要です。応答対象の要求の順序番号を指定します。

lua_rh 要求を送信する場合、*lua_rh* ビットの大部分は、送信するメッセージの RH (要求ヘッダー) と対応するように設定します。*lua_rh.pi* と *lua_rh.qri* は設定しないでください。この 2 つは、LUA によって設定されます。

応答を送信する場合は、次の 2 つの *lua_rh* ビットだけが使用されます。その他のビットは 0 (ゼロ) に設定してください。使用する *lua_rh* ビットは次のとおりです。

lua_rh.rrl

応答を示す 1 に設定します。

lua_rh.ri

肯定応答の場合は 0 に、否定応答の場合は 1 に設定します。

lua_flag1 パラメーター

次のフラグのうちのどれかを 1 に設定して、どのメッセージ・フローでデータを送信するかを指定してください。

lua_flag1.lu_exp

lua_flag1.sscp_norm

lua_flag1.lu_norm

これらのフラグのうちの 1 つだけを 1 に設定してください。CS/AIX では、アプリケーションは SSCP 急送フロー (*lua_flag1.sscp_exp* フラグ) でデータを送信することはできません。

lua_message_type

送信されるメッセージのタイプ。値は次のとおりです。

LUA_MESSAGE_TYPE_LU_DATA

LUA_MESSAGE_TYPE_SSCP_DATA

LUA_MESSAGE_TYPE_RSP

LUA_MESSAGE_TYPE_BID

LUA_MESSAGE_TYPE_BIS

LUA_MESSAGE_TYPE_CANCEL

LUA_MESSAGE_TYPE_CHASE

LUA_MESSAGE_TYPE_LUSTAT_LU

LUA_MESSAGE_TYPE_LUSTAT_SSCP

LUA_MESSAGE_TYPE_QC

LUA_MESSAGE_TYPE_QEC

LUA_MESSAGE_TYPE_RELQ

LUA_MESSAGE_TYPE_RTR

LUA_MESSAGE_TYPE_SBI

戻りパラメーター

LUA は常に次のパラメーターを戻します。

lua_flag2.async

このフラグは、*verb* が非同期で完了した場合は 1 に設定され、その *verb* が同期して完了した場合は 0 (ゼロ) に設定されます。

その他の戻りパラメーターは、*verb* が正常に完了したかどうかによって異なります。以下の節を参照してください。

正常な実行

verb が正常に実行された場合、LUA は次のパラメーターを戻します。

lua_prim_rc

LUA_OK

lua_sid この *verb* の発行時に、アプリケーションがセッション ID ではなく *lua_luname* パラメーターを指定した場合は、LUA はセッション ID を戻します。

lua_th 書き込まれたメッセージの完了 TH (LUA によって指定されたフィールドを含む)。ホストからの応答との相関のために、*lua_th.snf* (順序番号) の値を保管しなければならないことがあります。

lua_flag2 パラメーター

次のフラグのうちのどれかが 1 に設定され、どのメッセージ・フローでデータが送信されたかが示されます。

lua_flag2.lu_exp

lua_flag2.sscp_norm

lua_flag2.lu_norm

CS/AIX にインプリメントされた LUA では、アプリケーションは SSCP 急送フローでデータを送信することはできません。したがって、*sscp_exp* フラグが設定されることはありません (他のシステムにインプリメントされた LUA によって設定されることはあります)。

lua_sequence_number

LUA がデータ (データに RU のチェーンが必要な場合は最初の RU) の送信に使用する RU の順序番号。これは行形式で保管されます。

正常な実行: 状況情報

verb から LUA 状況情報が戻されると、LUA は次のパラメーターを戻します。

lua_prim_rc

LUA_STATUS

*lua_sec_rc***LUA_READY**

SLI セッションは、追加のコマンドを処理する準備ができています。この状況情報が使用されるのは、直前の LUA_NOT_READY 状況が報告された後か、あるいは、*lua_prim_rc* を LUA_CANCELLED に、*lua_sec_rc* を RECEIVED_UNBIND_HOLD または RECEIVED_UNBIND_NORMAL に設定して、SLI_CLOSE *verb* が完了した後です。

LUA_NOT_READY

SLI セッションが、次のいずれかの理由で一時的に中断状態になっています。

- CLEAR コマンドを受信した。SDT コマンドを受信すると、セッションは再開します。
- タイプ X'02' (BIND 受信予定) の UNBIND コマンドを受信した。セッションは、BIND、オプションの CRV および STSN、そして SDT などのコマンドを受信するまで中断状態になり、SDT を受信すると再開します。元の SLI_OPEN verb が提供したどのユーザー拡張ルーチンも再度呼び出されます。
- タイプ X'01' (通常) の UNBIND コマンドを受信し、このセッションの SLI_OPEN verb に *lua_session_type* LUA_SESSION_TYPE_DEDICATED が指定されていた。セッションは、BIND、オプションの CRV および STSN、そして SDT などのコマンドを受信するまで中断状態になり、SDT を受信すると再開します。元の SLI_OPEN verb が提供したどのユーザー拡張ルーチンも再度呼び出されます。

セッションの再開時に、アプリケーションは、SLI_BID または SLI_RECEIVE を発行して、READY 状況を受け取る必要があります。セッション状況が LUA_NOT_READY であっても、SLI_SEND verb と SLI_RECEIVE verb を SSCP 通常フローのデータに対して発行し続けることができます。

LUA_INIT_COMPLETE

アプリケーションは、LUA_OPEN_TYPE_PRIM_SSCP タイプを指定して SLI_OPEN を発行し、すでに発行されていた RUL_INIT verb は完了しています。アプリケーションは、SSCP 通常フローのデータに対して SLI_SEND verb および SLI_RECEIVE verb を発行できるようになりました。

LUA_SESSION_END_REQUESTED

ホストがアプリケーションに対して、セッションのシャットダウンを要求する SHUTD コマンドを送信しました。アプリケーションは、セッションを閉じる準備ができ次第、SLI_CLOSE を発行する必要があります。

実行の失敗

verb が正常に完了しなかった場合、LUA は、エラーのタイプを示す 1 次戻りコードと、実行が成功しなかった理由の詳細を示す 2 次戻りコードを戻します。

verb の取り消し: 以下の戻りコードは、別の verb によって取り消されたためにその verb が正常に完了しなかったことを示します。

lua_prim_rc
LUA_CANCELLED

lua_sec_rc

LUA_TERMINATED

このセッションに対して SLI_CLOSE verb が発行されたため、verb は取り消されました。

パラメーターの検査: 以下の戻りコードは、指定パラメーターに誤りがあったために、その `verb` が正常に完了しなかったことを示します。

lua_prim_rc

LUA_PARAMETER_CHECK

lua_sec_rc

値は次のとおりです。

LUA_BAD_DATA_PTR

lua_data_ptr パラメーターに指定された値が有効ではありませんでした。

LUA_BAD_SESSION_ID

lua_sid パラメーターと一致するセッション ID をもつアクティブな LUA LU セッションはありませんでした。

LUA_INVALID_FLOW

複数の *lua_flag1* フロー・フラグが 1 に設定されました。フロー・フラグのいずれか 1 つだけを 1 に設定して、データを送信するセッション・フローを指定してください。

lua_flag1.sscp_exp フロー・フラグが設定され、SSCP 急送フローでのメッセージの送信が指定されました。CS/AIX では、アプリケーションはこのフローでデータを送信することはできません。

LUA_INVALID_MESSAGE_TYPE

lua_message_type パラメーターが有効な値に設定されていませんでした。

AIX、LINUX

LUA_INVALID_POST_HANDLE

lua_post_handle パラメーターが、コールバック・ルーチンを指す有効なポインターではありませんでした。

LUA_REQUIRED_FIELD_MISSING

lua_flag1 フロー・フラグが何も設定されていませんでした。フロー・フラグのいずれか 1 つだけを 1 に設定する必要があります。

LUA_RESERVED_FIELD_NOT_ZERO

`verb` レコードの予約フィールド、またはこの `verb` で使用されないパラメーターが、ゼロ以外の値に設定されていました。

LUA_VERB_LENGTH_INVALID

lua_verb_length パラメーターの値に、この `verb` に必要な `verb` レコードの長さより小さい値が指定されていました。

LUA_DATA_LENGTH_ERROR

アプリケーションは、SLI_SEND を使用して LUSTAT をホストに送りましたが、状況情報に必要な 4 バイトを提供しませんでした。

状態の検査: 以下の戻りコードは、verb が発行されたときのセッション状態では、その verb が無効であったことを示します。

lua_prim_rc

LUA_STATE_CHECK

lua_sec_rc

値は次のとおりです。

LUA_MAX_NUMBER_OF SENDS

アプリケーションで この verb を発行したときに、2 つの verb はすでに進行中でした。アプリケーションでは、同時に未解決にしておく SLI_SEND verb は最大 2 つまでですが、これらは異なるセッション・フロー内になければなりません。

LUA_NO_SLI_SESSION

この verb で指定された LU 名では SLI_OPEN verb がまだ正常に完了していないか、あるいはセッションに障害が発生しました。

LUA_SEND_ON_FLOW_PENDING

この verb で指定されたセッション・フローには、未解決状態の SLI_SEND がすでにあります (セッション・フローは、*lua_flag1* フロー・フラグのどれかを 1 に設定することによって指定します)。各セッション・フローで同時に未解決のままにしておく SLI_SEND は 1 つだけです。

その他の状態: 以下の戻りコードは、指定された verb レコードは有効であったが、verb が正常に完了しなかったことを示します。

lua_prim_rc

LUA_UNSUCCESSFUL

lua_sec_rc

値は次のとおりです。

LUA_INVALID_PROCESS

この verb を発行したオペレーティング・システムのプロセスが、このセッションの SLI_OPEN verb を発行したプロセスと同じではありませんでした。セッションで verb を発行できるのは、そのセッションを開始したプロセスだけです。

LUA_INVALID_SESSION_PARAMETERS

アプリケーションは、SLI_SEND を使用して、ホストから受け取った BIND メッセージに対する肯定応答を送信しました。しかし、CS/AIX ノードは指定された BIND パラメーターを受け入れることができず、ホストに対して否定応答を送信しました。CS/AIX で受け入れられる BIND プロファイルの詳細については、38 ページの『SNA 情報』を参照してください。

LUA_RSP_CORRELATION_ERROR

SLI_SEND を使って応答を送信するときに、(応答の対象である受信メッセージの順序番号を示す) *lua_th.snf* パラメーターに有効な値が指定されませんでした。

LUA_RU_LENGTH_ERROR

lua_data_length パラメーターに指定された値が有効ではありませんでした。LU 通常フローでデータを送信する場合、最大長はホストから送られた BIND で指定されます。それ以外のすべてのフローでは、最大長は 256 バイトです。

LUA_NAU_INOPERATIVE

必要な SNA コンポーネント (LUA LU など) がアクティブでないか、異常終了状態です。

LUA_NO_SESSION

リモート LU との SNA セッションがアクティブではありません。

LUA_SLI_LOGIC_ERROR

この戻りコードは、次のいずれかを示します。

- ホスト・システムが SNA プロトコルに違反している
- LUA 内で内部エラーが検出された

SNA のトレース機能をアクティブにして問題を再現し (必要であればシステム管理者に連絡してください)、ホストが正しいデータを送信しているかどうかを調べてください。これでもまだ問題が解決されない場合は、CS/AIX のサポート担当者に連絡してください。

(その他の値)

その他の 2 次戻りコードはすべて、提供された SNA データが無効であったか、あるいは送信できなかったことを示す、SNA センス・コードです。戻される SNA センス・コードの解釈については、38 ページの『SNA 情報』を参照してください。

以下の戻りコードは、*verb* がその他の理由で正常に完了しなかったことを示します。

lua_prim_rc

LUA_COMM_SUBSYSTEM_ABENDED

CS/AIX の必須ソフトウェア構成要素 (ノードなど) が終了または停止しています。必要であれば、システム管理者に連絡してください。

lua_prim_rc

LUA_COMM_SUBSYSTEM_NOT_LOADED

Remote API Client ソフトウェアが開始されていないか、ノードが始動されていない、あるいは LUA アプリケーション向けに適切に構成されていません。CS/AIX LUA 構成パラメーターを調べて、Remote API Client とノードを始動してから、アプリケーションを実行してください。

lua_prim_rc

LUA_SESSION_FAILURE

LUA セッションに障害が発生しました。アプリケーションは、SLI_OPEN を再発行することによって、セッションを再開できます。

lua_sec_rc

値は次のとおりです。

LUA_LU_COMPONENT_DISCONNECTED

この戻りコードは、通信リンクまたはホスト LU の問題のために LUA セッションに障害が発生したことを示します。

LUA_RECEIVED_UNBIND

この戻りコードは、UNBIND コマンドがホストから送信され、セッションが終了したことを示します。この値は、セッションに対する SLI_OPEN verb に *lua_session_type* LUA_SESSION_TYPE_DEDICATED が指定されている場合にのみ生成されます。

LUA_SLI_LOGIC_ERROR

この戻りコードは、次のいずれかを示します。

- ホスト・システムが SNA プロトコルに違反している
- LUA 内で内部エラーが検出された

SNA のトレース機能をアクティブにして問題を再現し (必要であればシステム管理者に連絡してください)、ホストが正しいデータを送信しているかどうかを調べてください。それでもまだ問題が解決されない場合は、CS/AIX のサポート担当者に連絡してください。

lua_prim_rc

LUA_INVALID_VERB

lua_verb パラメーターも *lua_opcode* パラメーターも有効ではありませんでした。verb は実行されませんでした。

lua_prim_rc

LUA_STACK_TOO_SMALL

アプリケーションのスタック・サイズが小さすぎて、LUA が要求を完了できません。アプリケーションのスタック・サイズを大きくしてください。

lua_prim_rc

LUA_UNEXPECTED_DOS_ERROR

オペレーティング・システムでエラーが発生しました。

lua_sec_rc

この値はオペレーティング・システムからの戻りコードです。ご使用のオペレーティング・システムの資料で、この戻りコードの意味を調べてください。

他の verb との関連

この verb を発行するには、SLI_OPEN verb が正常に完了している必要があります。

既存の SLI_SEND が保留状態のときに、さらに SLI_SEND を発行するには、保留中の SLI_SEND とは別のセッション・フローを指定することが条件となります。すなわち、同じセッション・フローに対して複数の SLI_SEND を未解決にすることはできません。合計で 2 つより多くの SLI_SEND を未解決にすることはできません。

SSCP へのアクセスによる最初のセッション開始を指定する SLI_OPEN verb が正常に完了した後であればいつでも、SLI_SEND verb を SSCP 通常フローに対して発行できます。他のフローの SLI_SEND verb または他のセッション開始タイプの SLI_SEND verb は、BIND が受信された後でのみ許可され、BIND に指定されたプロトコルに従う必要があります。

使用法と制約事項

表 2 は、送信される SNA メッセージのタイプごとに、SLI_SEND のさまざまなパラメーターに有効な設定値を示しています。

表 2. メッセージ・タイプに応じた SLI_SEND パラメーターの設定値

SLI_SEND パラメーター	LU_DATA, SSCP_DATA	RSP	BID, BIS, RTR	CHASE QC	QEC, RELQ, SBL, SIG	RQR	LUSTAT_LU, LUSTAT_SSCP
<i>lua_rh</i>	FI, DR1I, DR2I, RI, BBI, EBI, CDI, CSI, EDI	RI	SDI, QRI	SDI, QRI, EBI, CDI	SDI	0	SDI, QRI, DR1I, DR2I, RI, BBI, EBI, CDI
<i>lua_th</i>	0	SNF	0	0	0	0	0
<i>lua_data_ptr</i>	必要 (データが ない場合はヌ ル)	必要 (データが ない場合はヌ ル)	ヌル	ヌル	ヌル	ヌル	必要
<i>lua_data_length</i>	必要	必要 (データが ない場合は 0)	0	0	0	0	必要
<i>lua_flag1</i> フロ ー・フラグ	0	必要 (1 に設 定)	0	0	0	0	0

アプリケーションは、SNA 応答を送信するときに以下の処理を行います。LUA は、提供された順序番号に基づいて、該当する要求コードを入れます。

- *lua_message_type* を LUA_MESSAGE_TYPE_RSP に設定する。
- *lua_th.snf* を応答先の要求の順序番号に設定する。
- 該当する *lua_flag1* フロー・フラグを設定する。
- 要求コードのみ必要な肯定応答の場合は、*lua_rh.ri* および *lua_data_length* のいずれも 0 (ゼロ) に設定する。
- 否定応答の場合は、次のようにする。
 - *lua_rh.ri* を 1 に設定する。
 - *lua_data_ptr* を該当する SNA センス・コードを指すように設定する。
 - *lua_data_length* を 4 (センス・コードの長さ) に設定する。

SLI_SEND の正常終了は、メッセージが正常にデータ・リンクのキューに入ったことを示します。メッセージが正常に送信されたことや、ホストがメッセージを受け入れたことを示すとは限りません。

CS/AIX LU または ホスト LU の処理能力を超える量のデータを LUA アプリケーションが送信しないようにするために、1 次と 2 次間のハーフ・セッション (これは BIND で指定されます) でペーシングを使用することができます。その場合、LU 通常フローに対する SLI_SEND が遅れたり、完了に時間がかかることがあります。

SLI_BIND_ROUTINE

この verb は、LUA からアプリケーションへ送信されるもので (アプリケーションが SLI_OPEN verb に指定する BIND 拡張ルーチンのエントリー・ポイントを使用)、アプリケーションから LUA へは送信されません。

SLI_BIND_ROUTINE verb は、ホストからの BIND 要求を LUA アプリケーションに渡します。アプリケーションは、BIND をそのままの状態を受け入れるか、BIND パラメーターの折衝を試みる際に変更する、あるいは適切な SNA センス・コードを使用してリジェクトすることができます。

指定パラメーター

LUA は、以下のパラメーターをアプリケーションに提供します。

lua_verb

LUA_VERB_SLI

lua_verb_length

LUA verb レコードの長さ (バイト数)。

lua_opcode

LUA_OPCODE_SLI_BIND_ROUTINE

lua_luname

セッションで使用する LU の ASCII 文字の名前。

lua_sid セッションのセッション ID。

lua_data_length

提供される BIND RU の長さ。

lua_data_ptr

提供される BIND RU を格納するバッファを指すポインター。

lua_th BIND からの TH パラメーター。

lua_rh BIND からの RH パラメーター。

戻りパラメーター

アプリケーションから戻されるパラメーターは、verb が正常に完了したかどうかによって異なります。以下の節を参照してください。

正常な実行: 受け入れ済みまたは折衝された BIND

アプリケーションは、BIND を受け入れるか折衝することにした場合、以下のパラメーターを戻します。

lua_prim_rc

LUA_OK

lua_data_ptr

提供される BIND RU を格納するバッファを指すポインター。アプリケーションが BIND をそのままの状態を受け入れる場合、バッファの内容を変更してはなりません。BIND 内で 1 つ以上のパラメーターの折衝を試みる場合は、データを変更して、該当のパラメーターを希望する値に設定する必要があります。

SLI_BIND_ROUTINE

正常でない実行: BIND のリジェクト

アプリケーションは、BIND をリジェクトすることにした場合、以下のパラメーターを戻します。

lua_prim_rc

LUA_NEGATIVE_RSP

lua_data_length

戻された SNA センス・コードの長さ (*lua_data_ptr* パラメーター内)。

lua_data_ptr

バッファーへのポインター。このバッファーには、アプリケーションが BIND をリジェクトした理由に関連した SNA センス・コードが入っています。

他の verb との関連

LUA は、SLI_OPEN verb の処理中 (アプリケーションが SLI_OPEN を発行してから、それが非同期で戻るまで) にこのルーチンを呼び出します。

使用法と制約事項

アプリケーションの拡張ルーチンには非同期で戻るメカニズムはありません。ルーチンは同期で戻る必要があります。

SLI_SDT_ROUTINE

この verb は、LUA からアプリケーションへ送信されるもので (アプリケーションが SLI_OPEN verb に指定する SDT 拡張ルーチンのエントリー・ポイントを使用)、アプリケーションから LUA へは送信されません。

SLI_SDT_ROUTINE verb は、ホストからの SDT 要求を LUA アプリケーションに渡します。アプリケーションは、SDT 応答で応えるか、該当する SNA センス・コードを使用してリジェクトします。

指定パラメーター

LUA は、以下のパラメーターをアプリケーションに提供します。

lua_verb

LUA_VERB_SLI

lua_verb_length

LUA verb レコードの長さ (バイト数)。

lua_opcode

LUA_OPCODE_SLI_SDT_ROUTINE

lua_luname

セッションで使用する LU の ASCII 文字の名前。

lua_sid セッションのセッション ID。

lua_data_length

提供される SDT RU の長さ。

lua_data_ptr

提供される SDT RU を格納するバッファを指すポインタ。

lua_th SDT からの TH パラメータ。

lua_rh SDT からの RH パラメータ。

戻りパラメータ

アプリケーションから戻されるパラメータは、*verb* が正常に完了したかどうかによって異なります。以下の節を参照してください。

正常な実行: SDT 応答

アプリケーションは、SDT を受け入れることにした場合、以下のパラメータを戻します。

lua_prim_rc

LUA_OK

lua_data_ptr

提供される SDT 応答 RU を含むバッファを指すポインタ。

正常でない実行: SDT のリジェクト

アプリケーションは、SDT をリジェクトすることにした場合、以下のパラメータを戻します。

lua_prim_rc

LUA_NEGATIVE_RSP

lua_data_length

戻された SNA センス・コードの長さ (*lua_data_ptr* パラメータ内)。

lua_data_ptr

バッファへのポインタ。このバッファには、アプリケーションが SDT をリジェクトした理由に関連した SNA センス・コードが入っています。

他の *verb* との関連

LUA は、SLI_OPEN *verb* の処理中 (アプリケーションが SLI_OPEN を発行してから、それが非同期で戻るまで) にこのルーチンを呼び出します。

使用法と制約事項

アプリケーションの拡張ルーチンには非同期で戻るメカニズムはありません。ルーチンは同期で戻る必要があります。

SLI_STSN_ROUTINE

この *verb* は、LUA からアプリケーションへ送信されるもので (アプリケーションが SLI_OPEN *verb* に指定する STSN 拡張ルーチンのエントリ・ポイントを使用)、アプリケーションから LUA へは送信されません。

SLI_STSN_ROUTINE

SLI_STSN_ROUTINE verb は、ホストからの STSN 要求を LUA アプリケーションに渡します。アプリケーションは、STSN 応答で応えるか、該当する SNA センス・コードを使用してリジェクトします。

指定パラメーター

LUA は、以下のパラメーターをアプリケーションに提供します。

lua_verb

LUA_VERB_SLI

lua_verb_length

LUA verb レコードの長さ (バイト数)。

lua_opcode

LUA_OPCODE_SLI_STSN_ROUTINE

lua_luname

セッションで使用する LU の ASCII 文字の名前。

lua_sid セッションのセッション ID。

lua_data_length

提供される STSN RU の長さ。

lua_data_ptr

提供される STSN RU を格納するバッファーを指すポインター。

lua_th STSN からの TH パラメーター。

lua_rh STSN からの RH パラメーター。

戻りパラメーター

アプリケーションから戻されるパラメーターは、verb が正常に完了したかどうかによって異なります。以下の節を参照してください。

正常な実行: STSN 応答

アプリケーションは、STSN を受け入れることにした場合、以下のパラメーターを戻します。

lua_prim_rc

LUA_OK

lua_data_ptr

提供される STSN 応答 RU を含むバッファーを指すポインター。

正常でない実行: STSN のリジェクト

アプリケーションは、STSN をリジェクトすることにした場合、以下のパラメーターを戻します。

lua_prim_rc

LUA_NEGATIVE_RSP

lua_data_length

戻された SNA センス・コードの長さ (*lua_data_ptr* パラメーター内)。

lua_data_ptr

バッファへのポインタ。このバッファには、アプリケーションが STSN をリジェクトした理由に関連した SNA センス・コードが入っています。

他の verb との関連

LUA は、SLI_OPEN verb の処理中 (アプリケーションが SLI_OPEN を発行してから、それが非同期で戻るまで) にこのルーチンを呼び出します。

使用法と制約事項

アプリケーションの拡張ルーチンには非同期で戻るメカニズムはありません。ルーチンは同期で戻る必要があります。

第 6 章 LUA アプリケーションのサンプル

この章では、AIX または Linux オペレーティング・システム用に作成された CS/AIX サンプル LUA プログラム `lsample.c` について説明します。ここでは、LUA RUI verbs の使用法を示しています。このファイルは、ディレクトリー `/usr/lib/sna/samples` (AIX) または `/opt/ibm/sna/samples` (Linux) に格納されます。

ここでは、次の事項について説明します。

- アプリケーションの処理の概要
- アプリケーションのコンパイル、リンク、実行の手順

処理の概要

このアプリケーションは、非常に単純な 3270 エミュレーション・プログラムです。このアプリケーションは、(LU セッションと SSCP セッションの両方で) ホストから送信される画面データの不定様式表示と、(アプリケーションが LU セッションに接続されたか SSCP セッションに接続されたかを示す) 状況メッセージとを提供します。ホストから確定応答要求を受け取ると、自動的に肯定応答が送信されます。ユーザーが入力するデータは、ホストに送信されます。ただし、次の 2 つの特殊キー・ストロークは例外です。

[(左大括弧)

LU セッションと SSCP セッションとの切り替え

] (右大括弧)

アプリケーションの終了

このプログラムは、初期化処理とその後の 2 つの必須のメイン・ループから構成されています。1 つはホストからデータを読み取るループ、もう 1 つはユーザーから提供されたデータをホストに送信するループです。この 2 つのメイン・ループは、次のようにインプリメントされています。

読み取りループでは、RUI_READ verb の再帰呼び出しを使用します。コールバック・ルーチンでは、次に示す処理が実行されます (コールバック・ルーチンは、verb が非同期で完了した場合に LUA が呼び出すルーチンです)。

- 画面データをすべて画面に書き込む
- セッション状況情報をすべて処理する
- 応答が必要な場合は、肯定応答を作成して送信する
- その後、RUI_READ verb を再発行して、ループを続行する

verb が同期して完了した場合は、コールバック・ルーチンとして使用されるのと同じルーチンが、戻り時に明示的に呼び出されます。これにより、戻りのタイプに関係なく、同じ処理が実行されます。

書き込みループでは、キーボードからのデータを読み取ります。上記の 2 つの特殊キー・ストロークのいずれかが指定されると、その機能が有効になります。それ以外の場合は、着信データは (CSV CONVERT verb を使って) 拡張 2 進化 10 進コ

処理の概要

ード (EBCDIC) に変換され、LU セッションまたは SSCP セッションのどちらかでホストに送信されます。どちらのセッションかは、アプリケーションがその時点でどちらに接続されているかによって決まります。データは RUI_WRITE verb を使って送信されます。このループでも、verb が非同期で完了するかどうかに関係なく、コールバック・ルーチンが使用されます。このプログラムは、コールバック・ルーチンによってセマフォがクリアされるのを待ってから、ループを続行します。

ユーザーが] (右大括弧) キー・ストロークを入力してアプリケーションを終了しようとする、プログラムは書き込みループから出て、RUI_TERM verb を発行してセッションを終了します。読み取りループで RUI_READ verb から LUA_OK 以外の戻りコードが検出された場合も、セッションは終了します。

プログラムの流れを 図 5 のダイアグラムで示します。

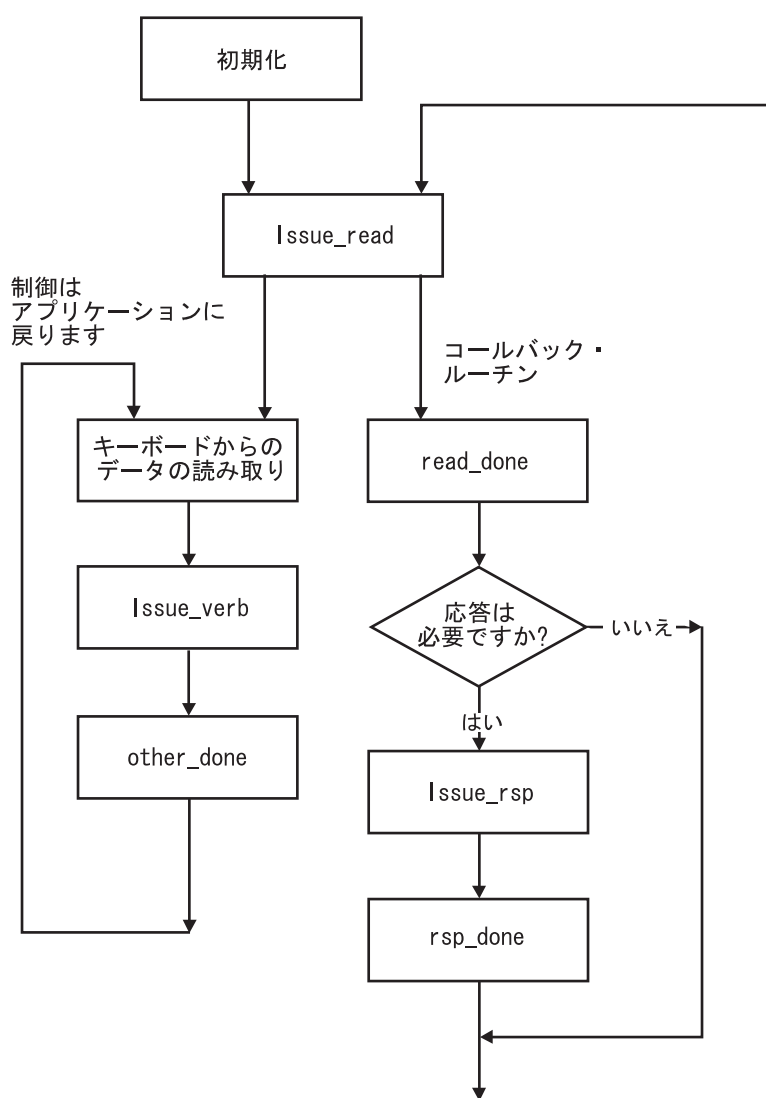


図 5. サンプルの LUA アプリケーションのプログラムの流れ

アプリケーションのテスト

サンプル・アプリケーションのソース・コードを調べた後で、それをテストできます。必要な手順は次のとおりです。

1. アプリケーションの実行に適したホスト・コンピューターにアクセスできることを確認する。
2. アプリケーションをコンパイルしてリンクする。
3. LUA で使用する CS/AIX を構成する (この作業は通常、システム管理者が実行します)。
4. アプリケーションを実行する。

上記手順については、以下の節で詳しく説明します。

ホスト要件

サンプル・アプリケーションを実行するには、ホスト・コンピューターに LU が必要です。このサンプル・アプリケーションは 3270 ディスプレイ端末をエミュレートするため、LU はホスト上で 3278 や 3279 などの 3270 ディスプレイ LU (LU タイプ 2) として構成されていなければなりません。CS/AIX で LU を構成するときは、ホストで割り当てられた LU 番号を使用する必要があります。

サンプル・アプリケーションの構成

CS/AIX は、必要な LU を組み込むように構成する必要があります。この作業は通常、システム管理者によって実行されます。必要なコンポーネントは次のとおりです。

- DLC、ポート、LS。
- タイプ 0 から 3 の LU。ホスト上の適切な LU の番号と一致する LU 番号。

これらのコンポーネントには、自由に名前を付けられます。アプリケーションが必要とする情報は、セッションに使用する LU のみです。この情報は、1 つのコマンド行パラメーター (LU 名) として、あるいは 2 つのコマンド行パラメーター (PU 名と LU 番号) として、アプリケーションに渡されます。LU の構成には、次の項目も適用されます。

- CS/AIX 構成情報の中でこの LU に対して構成されている LU 番号は、ホストで割り当てられた LU 番号と一致していなければなりません。
- アプリケーションで使用するために、1 つ以上の LU から成る LU プールを構成できます。プールにアクセスするには、プールの名前を指定するか、そのプール内の LU の名前を指定します。プール内の最初に使用可能な LU が使用されます。

サンプル・アプリケーションのコンパイルとリンク

AIX または Linux システムでプログラムをコンパイルしてリンクするには、以下のステップを実行します。

1. ファイル `lsample.c` をディレクトリー `/usr/lib/sna/samples` からプライベート・ディレクトリーにコピーする。
2. AIX でプログラムをコンパイルしてリンクするには、次のコマンドを使用します。

アプリケーションのテスト

```
cc -o lsample -I /usr/include/sna -bimport:/usr/lib/sna/lua.exp -bimport:/usr/lib/sna/csv.exp lsample.c
```

Linux でプログラムをコンパイルしてリンクするには、次のコマンドを使用します。

```
gcc -o lsample -I /opt/ibm/sna/include -L /opt/ibm/sna/lib -llua -lsna -lcsv -lpLiS -lpthread lsample.c
```

サンプル・アプリケーションの実行

この項では、173 ページの『サンプル・アプリケーションのコンパイルとリンク』の説明に従ってサンプル・アプリケーションのコンパイルとリンクが完了しているものとして説明します。

サンプル・アプリケーションは、LUA の他に CSV インターフェースも使用します。このサンプル・アプリケーションには、CSV CONVERT verb への呼び出しが含まれています。この verb は、ユーザー提供のデータをホストに送信する前に ASCII から EBCDIC に変換し、ホストから受信したデータを画面に表示する前に ASCII に変換します。この変換では、ユーザー定義の変換テーブル (Table G) が使用されます。このテーブルは CS/AIX コンピューター上のファイルに保管されます。この LUA サンプル・アプリケーション・プログラム・ソースで、適切なファイル **luatblg.dat** がディレクトリー **/usr/lib/sna/samples** (AIX) または **/opt/ibm/sna/samples** (Linux) に提供されます。

サンプル・アプリケーションを実行するには、次の手順に従ってください。

1. CS/AIX ソフトウェアが起動されていることと、ホストへの LS がアクティブであることを確認します。必要であれば、システム管理者に連絡してください。
2. 環境変数 SNATBLG を、Table G 変換テーブルが入っているファイルの名前に設定します。そのファイルが現行ディレクトリーにない場合は、絶対パスを指定してください。
3. 次のコマンドのいずれかを入力して、アプリケーションを起動します。

```
lsample luname
```

```
lsample puname lunumber
```

この例では、*luname* にはこのアプリケーション用に構成した LU 名 (あるいは、LU プールの名前、またはプール内の LU 名) を指定します。

この例では、*puname* には必要な LU を所有している PU の名前を指定し、*lunumber* には LU 番号を (10 進数で) 指定します。

アプリケーションは、ホストとのセッションの確立に成功すると、メッセージ「**LU active (LU アクティブ)**」を表示します。

4. 通常どおりにデータを入力して、ログオンし、ホスト・アプリケーションにアクセスします。
5. LU セッションと SSCP セッションとを切り替えるには、[(左大括弧) キーを押して、**Enter** キーを押します。

アプリケーションは、メッセージ「**LU session (LU セッション)**」または「**SSCP session (SSCP セッション)**」を表示して、現在どちらのセッションに接続しているかを示します。また、**BIND** または **UNBIND** メッセージを受信したときには自動的に切り替えが行われます。

6. ホスト・アプリケーションでの作業が終わったら、通常の手順に従ってホスト・アプリケーションを終了し、ログオフします。
7. サンプル・アプリケーションを終了するには、] (右大括弧) キーを押して、**Enter** キーを押します。

アプリケーションは、メッセージ「**Closedown (作業停止)**」を表示した後、「**Terminated (終了)**」を表示し、ホストとのセッションが終了したことを知らせます。メッセージ「**Read failed (読み取り失敗)**」と、未解決の **RUI_READ verb** が **RUI_TERM verb** によって取り消されたことを示す戻りコードが表示されることもあります。

付録 A. 戻りコードの値

この付録は、LUA インターフェースのすべての可能な戻りコードを番号順にリストしています。値は、LUA ヘッダー・ファイル **lua_c.h** (AIX または Linux の場合) または **winlua.h** (Windows の場合) に定義されています。

この付録をリファレンスとして使用して、アプリケーションが受け取った戻りコードの意味を検査することができます。

1 次戻りコード

LUA アプリケーションでは以下の 1 次戻りコードが使用されます。

LUA_OK	0x0100
LUA_STATE_CHECK	0x0200
LUA_COMM_SUBSYSTEM_ABENDED	0x03F0
LUA_COMM_SUBSYSTEM_NOT_LOADED	0x04F0
LUA_INVALID_VERB_SEGMENT	0x08F0
LUA_SESSION_FAILURE	0x0FF0
LUA_UNEXPECTED_DOS_ERROR	0x11F0
LUA_UNSUCCESSFUL	0x1400
LUA_STACK_TOO_SMALL	0x15F0
LUA_NEGATIVE_RSP	0x1800
LUA_CANCELLED	0x2100
LUA_IN_PROGRESS	0x3000
LUA_STATUS	0x4000
LUA_INVALID_VERB	0xFFFF

2 次戻りコード

LUA アプリケーションでは以下の 2 次戻りコードが使用されます。

LUA_SEC_RC_OK	0x00000000
LUA_INVALID_LUNAME	0x01000000
LUA_BAD_SESSION_ID	0x02000000
LUA_DATA_TRUNCATED	0x03000000
LUA_BAD_DATA_PTR	0x04000000
LUA_DATA_SEG_LENGTH_ERROR	0x05000000
LUA_RESERVED_FIELD_NOT_ZERO	0x06000000
LUA_INVALID_POST_HANDLE	0x07000000
LUA_PURGED	0x0C000000
LUA_BID_VERB_SEG_ERROR	0x0F000000
LUA_NO_PREVIOUS_BID_ENABLED	0x10000000
LUA_NO_DATA	0x11000000
LUA_BID_ALREADY_ENABLED	0x12000000
LUA_VERB_RECORD_SPANS_SEGMENTS	0x13000000
LUA_INVALID_FLOW	0x14000000
LUA_NOT_ACTIVE	0x15000000
LUA_VERB_LENGTH_INVALID	0x16000000
LUA_REQUIRED_FIELD_MISSING	0x19000000
LUA_READY	0x30000000
LUA_NOT_READY	0x31000000
LUA_INIT_COMPLETE	0x32000000
LUA_SESSION_END_REQUESTED	0x33000000
LUA_NO_SLI_SESSION	0x34000000
LUA_SESSION_ALREADY_OPEN	0x35000000
LUA_INVALID_OPEN_INIT_TYPE	0x36000000
LUA_INVALID_OPEN_DATA	0x37000000

2 次戻りコード

LUA_UNEXPECTED_SNA_SEQUENCE	0x38000000
LUA_NEG_RSP_FROM_BIND_ROUTINE	0x39000000
LUA_NEG_RSP_FROM_CRV_ROUTINE	0x3A000000
LUA_NEG_RSP_FROM_STSN_ROUTINE	0x3B000000
LUA_CRV_ROUTINE_REQUIRED	0x3C000000
LUA_STSN_ROUTINE_REQUIRED	0x3D000000
LUA_INVALID_OPEN_ROUTINE_TYPE	0x3E000000
LUA_MAX_NUMBER_OF SENDS	0x3F000000
LUA_SEND_ON_FLOW_PENDING	0x40000000
LUA_INVALID_MESSAGE_TYPE	0x41000000
LUA_RECEIVE_ON_FLOW_PENDING	0x42000000
LUA_DATA_LENGTH_ERROR	0x43000000
LUA_CLOSE_PENDING	0x44000000
LUA_NEGATIVE_RSP_CHASE	0x46000000
LUA_NEGATIVE_RSP_SHUTC	0x47000000
LUA_NEGATIVE_RSP_RSHUTD	0x48000000
LUA_NO_RECEIVE_TO_PURGE	0x4A000000
LUA_CANCEL_COMMAND_RECEIVED	0x4D000000
LUA_RUI_WRITE_FAILURE	0x4E000000
LUA_INVALID_SESSION_TYPE	0x4F000000
LUA_SLI_BID_PENDING	0x51000000
LUA_SLI_PURGE_PENDING	0x52000000
LUA_PROCEDURE_ERROR	0x53000000
LUA_INVALID_SLI_ENCR_OPTION	0x54000000
LUA_RECEIVED_UNBIND	0x55000000
LUA_RECEIVED_UNBIND_HOLD	0x56000000
LUA_RECEIVED_UNBIND_NORMAL	0x57000000
LUA_SLI_LOGIC_ERROR	0x7F000000
LUA_TERMINATED	0x80000000
LUA_NO_RUI_SESSION	0x81000000
LUA_DUPLICATE_RUI_INIT	0x82000000
LUA_INVALID_PROCESS	0x83000000
LUA_API_MODE_CHANGE	0x85000000
LUA_COMMAND_COUNT_ERROR	0x87000000
LUA_NO_READ_TO_PURGE	0x88000000
LUA_MULTIPLE_WRITE_FLOWS	0x89000000
LUA_DUPLICATE_READ_FLOW	0x8A000000
LUA_DUPLICATE_WRITE_FLOW	0x8B000000
LUA_LINK_NOT_STARTED	0x8C000000
LUA_INVALID_ADAPTER	0x8D000000
LUA_ENCR_DECR_LOAD_ERROR	0x8E000000
LUA_ENCR_DECR_PROC_ERROR	0x8F000000
LUA_INVALID_PUNAME	0x90000000
LUA_UNAUTHORIZED_ACCESS	0x90020000
LUA_INVALID_LUNUMBER	0x91000000
LUA_INVALID_FORMAT	0x92000000
LUA_DUPLICATE_RUI_REINIT	0x93000000
LUA_REINIT_INVALID	0x94000000
LUA_TCPCV_LENGTH_INVALID	0x95000000
LUA_LINK_NOT_STARTED_RETRY	0x95FF0000
LUA_NEG_RSP_FROM_SDT_ROUTINE	0x96000000
LUA_NEG_NOTIFY_RSP	0xBE000000
LUA_RUI_LOGIC_ERROR	0xBF000000
LUA_COBOL_NOT_SUPPORTED	0xC0000000
LUA_DUPLICATE_RUI_INIT_PRIMARY	0xC2000000
LUA_LU_INOPERATIVE	0xFF000000

以下の 2 次戻りコードは SNA センス・コードです。それらを、LUA が使用する標準の標準のバイト配列と、SNA 解説書で SNA センス・コードに使用されるバイト配列の両方で示します。

LUA_RESOURCE_NOT_AVAILABLE	0x00000108	(SNA sense 0801 0000)
LUA_RU_DATA_ERROR	0x00000110	(SNA sense 1001 0000)
LUA_INCORRECT_SEQUENCE_NUMBER	0x00000120	(SNA sense 2001 0000)
LUA_INVALID_SC_OR_NC_RH	0x00000140	(SNA sense 4001 0000)
LUA_RU_LENGTH_ERROR	0x00000210	(SNA sense 1002 0000)

LUA_CHAINING_ERROR	0x00000220	(SNA sense 2002 0000)
LUA_FUNCTION_NOT_SUPPORTED	0x00000310	(SNA sense 1003 0000)
LUA_BRACKET	0x00000320	(SNA sense 2003 0000)
LUA_BB_NOT_ALLOWED	0x00000340	(SNA sense 4003 0000)
LUA_NAU_INOPERATIVE	0x00000380	(SNA sense 8003 0000)
LUA_DIRECTION	0x00000420	(SNA sense 2004 0000)
LUA_EB_NOT_ALLOWED	0x00000440	(SNA sense 4004 0000)
LUA_SESSION_LIMIT_EXCEEDED	0x00000508	(SNA sense 0805 0000)
LUA_DATA_TRAFFIC_RESET	0x00000520	(SNA sense 2005 0000)
LUA_NO_SESSION	0x00000580	(SNA sense 8005 0000)
LUA_DATA_TRAFFIC_QUIESCED	0x00000620	(SNA sense 2006 0000)
LUA_EXCEPTION_RSP_NOT_ALLOWED	0x00000640	(SNA sense 4006 0000)
LUA_CATEGORY_NOT_SUPPORTED	0x00000710	(SNA sense 1007 0000)
LUA_DATA_TRAFFIC_NOT_RESET	0x00000720	(SNA sense 2007 0000)
LUA_DEFINITE_RSP_NOT_ALLOWED	0x00000740	(SNA sense 4007 0000)
LUA_NO_BEGIN_BRACKET	0x00000820	(SNA sense 2008 0000)
LUA_PACING_NOT_SUPPORTED	0x00000840	(SNA sense 4008 0000)
LUA_MODE_INCONSISTENCY	0x00000908	(SNA sense 0809 0000)
LUA_SC_PROTOCOL_VIOLATION	0x00000920	(SNA sense 2009 0000)
LUA_CD_NOT_ALLOWED	0x00000940	(SNA sense 4009 0000)
LUA_IMMEDIATE_REQ_MODE_ERROR	0x00000A20	(SNA sense 200A 0000)
LUA_NO_RESPONSE_NOT_ALLOWED	0x00000A40	(SNA sense 400A 0000)
LUA_BRACKET_RACE_ERROR	0x00000B08	(SNA sense 800B 0000)
LUA_QUEUED_RESPONSE_ERROR	0x00000B20	(SNA sense 200B 0000)
LUA_CHAINING_NOT_SUPPORTED	0x00000B40	(SNA sense 400B 0000)
LUA_ERP_SYNC_EVENT_ERROR	0x00000C20	(SNA sense 200C 0000)
LUA_BRACKETS_NOT_SUPPORTED	0x00000C40	(SNA sense 400C 0000)
LUA_RSP_BEFORE_SENDING_REQ	0x00000D20	(SNA sense 200D 0000)
LUA_CD_NOT_SUPPORTED	0x00000D40	(SNA sense 400D 0000)
LUA_RSP_CORRELATION_ERROR	0x00000E20	(SNA sense 200E 0000)
LUA_RSP_PROTOCOL_ERROR	0x00000F20	(SNA sense 200F 0000)
LUA_INCORRECT_USE_OF_FI	0x00000F40	(SNA sense 400F 0000)
LUA_ALTERNATE_CODE_NOT_SUPPORT	0x00001040	(SNA sense 4001 0000)
LUA_INCORRECT_RU_CATEGORY	0x00001140	(SNA sense 4011 0000)
LUA_INSUFFICIENT_RESOURCES	0x00001208	(SNA sense 0812 0000)
LUA_INCORRECT_REQUEST_CODE	0x00001240	(SNA sense 4012 0000)
LUA_BB_REJECT_NO_RTR	0x00001308	(SNA sense 0813 0000)
LUA_INCORRECT_SPEC_OF_SDI_RTI	0x00001340	(SNA sense 4013 0000)
LUA_BB_REJECT_RTR	0x00001408	(SNA sense 0814 0000)
LUA_INCORRECT_DR1I_DR2I_ERI	0x00001440	(SNA sense 4014 0000)
LUA_INCORRECT_USE_OF_QRI	0x00001540	(SNA sense 4015 0000)
LUA_INCORRECT_USE_OF EDI	0x00001640	(SNA sense 4016 0000)
LUA_INCORRECT_USE_OF_PDI	0x00001740	(SNA sense 4017 0000)
LUA_RECEIVER_IN_TRANSMIT_MODE	0x00001B08	(SNA sense 081B 0000)
LUA_REQUEST_NOT_EXECUTABLE	0x00001C08	(SNA sense 081C 0000)
LUA_INVALID_SESSION_PARAMETERS	0x00002108	(SNA sense 0821 0000)
LUA_UNIT_OF_WORK_ABORTED	0x00002408	(SNA sense 0824 0000)
LUA_FM_FUNCTION_NOT_SUPPORTED	0x00002608	(SNA sense 0826 0000)
LUA_LU_COMPONENT_DISCONNECTED	0x00003108	(SNA sense 0831 0000)
LUA_INVALID_PARAMETER_FLAGS	0x00003308	(SNA sense 0833 0000)
LUA_INVALID_PARAMETER	0x00003508	(SNA sense 0835 0000)
LUA_CRYPTOGRAPHY_INOPERATIVE	0x00004808	(SNA sense 0848 0000)
LUA_REQ_RESOURCES_NOT_AVAIL	0x00004B08	(SNA sense 084B 0000)
LUA_SSCP_LU_SESSION_NOT_ACTIVE	0x00005708	(SNA sense 0857 0000)
LUA_SYNC_EVENT_RESPONSE	0x00006708	(SNA sense 0867 0000)
LUA_SESSION_SERVICE_PATH_ERROR	0x00007D08	(SNA sense 087D 0000)
LUA_NEGOTIABLE_BIND_ERROR	0x01003508	(SNA sense 0835 0001)
LUA_REC_CORR_TABLE_FULL	0x01007808	(SNA sense 0878 0001)
LUA_NON_UNIQ_ID	0x011000C0	(SNA sense C000 1001)
LUA_INV_NAU_ADDR	0x012000C0	(SNA sense C000 2001)
LUA_BIND_FM_PROFILE_ERROR	0x02003508	(SNA sense 0835 0002)
LUA_SSCP_PLU_SESS_NOT_ACTIVE	0x02005708	(SNA sense 0857 0002)
LUA_SEND_CORR_TABLE_FULL	0x02007808	(SNA sense 0878 0002)
LUA_NON_UNIQ_NAU_AD	0x021000C0	(SNA sense C000 1002)
LUA_INV_ADPT_NUM	0x022000C0	(SNA sense C000 2002)
LUA_BIND_TS_PROFILE_ERROR	0x03003508	(SNA sense 0835 0003)
LUA_SSCP_SLU_SESS_INACT	0x03005708	(SNA sense 0857 0003)

2 次戻りコード

LUA_SLU_SESSION_LIMIT_EXCEEDED	0x0A000508	(SNA sense 0805 000A)
LUA_BIND_LU_TYPE_ERROR	0x0E003508	(SNA sense 0835 000E)
LUA_HDX_BRACKET_STATE_ERROR	0x21010510	(SNA sense 1005 0121)
LUA_RESPONSE_ALREADY_SENT	0x22010510	(SNA sense 1005 0122)
LUA_EXR_SENSE_INCORRECT	0x23010510	(SNA sense 1005 0123)
LUA_RESPONSE_OUT_OF_ORDER	0x24010510	(SNA sense 1005 0124)
LUA_CHASE_RESPONSE_REQUIRED	0x25010510	(SNA sense 1005 0125)

付録 B. アクセシビリティ

アクセシビリティ機能は、運動障害または視覚障害など身体に障害を持つユーザーがソフトウェア・プロダクトを快適に使用できるようにサポートします。z/OS™ のアクセシビリティの主要機能により、ユーザーは以下のことができるようになります。

- スクリーン・リーダーおよび画面拡大機能などの支援テクノロジーの使用
- キーボードのみを使用して、特定の機能または画面を使用したのと同等の機能を操作
- 色、コントラスト、フォント・サイズなど表示属性のカスタマイズ

支援機能の使用

スクリーン・リーダーなどの支援テクノロジーは、z/OS のユーザー・インターフェースを使用して機能します。この支援テクノロジーを使用して z/OS インターフェースにアクセスする場合、その特定情報については支援テクノロジーの資料を参照してください。

ユーザー・インターフェースのキーボード・ナビゲーション

ユーザーは、TSO/E または ISPF を使用して z/OS ユーザー・インターフェースにアクセスできます。TSO/E インターフェースおよび ISPF インターフェースへのアクセスについての詳細は、「z/OS TSO/E 入門」(邦文番号 SA88-8632: 英文番号 SA22-7787)、「z/OS TSO/E ユーザーズ・ガイド」(邦文番号 SA88-8638: 英文番号 SA22-7794)、および「対話式システム生産性向上機能 (ISPF) ユーザーズ・ガイド 第 1 巻 z/OS バージョン 1 リリース 2.0」(邦文番号 SC88-8965: 英文番号 SC34-4822) を参照してください。上記の資料には、キーボード・ショートカットまたはファンクション・キー (PF キー) の使用方法を含む TSO/E および ISPF の使用方法が記載されています。それぞれの資料では、PF キーのデフォルトの設定値とそれらの機能の変更方法についても説明しています。

z/OS の情報

z/OS の情報は、スクリーン・リーダー (読み上げソフトウェア) を使用して、BookServer/Library Server バージョンの z/OS ブックを次のインターネット・ライブラリーでアクセスできます。

<http://www.ibm.com/servers/eserver/zseries/zos/bkserv/>

付録 C. 特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものであり、本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。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。

〒106-0032
東京都港区六本木 3-2-31
IBM World Trade Asia Corporation
Licensing

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。

IBM Corporation, Site Counsel
P.O. Box 12195
3039 Cornwallis Road
Research Triangle Park, NC 27709-2195
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せんが、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の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。

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。お客様は、IBM 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。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生した創作物にも、次のように、著作権表示を入れていただく必要があります。「© (お

お客様の会社名) (西暦年)」このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。 © IBM Corp. 2000, 2005. All rights reserved.

商標

以下は、IBM Corporation の商標です。

ACF/VTAM	IBM
Advanced Peer-to-Peer Networking	IBMLink
AIX	IMS
AIXwindows	MVS
AnyNet	MVS/ESA
Application System/400	Operating System/2
APPN	Operating System/400
AS/400	OS/2
CICS	OS/400
DATABASE 2	PowerPC
DB2	PowerPC Architecture
Enterprise System/3090	pSeries
Enterprise System/4381	S/390
Enterprise System/9000	System/390
ES/3090	VSE/ESA
ES/9000	VTAM
eServer	WebSphere
	zSeries

以下は、それぞれ各会社の商標または登録商標です。

Java およびすべての Java ベース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または登録商標です。

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。

Intel は、Intel Corporation の米国およびその他の国における商標です。

Linux は、Linus Torvalds の米国およびその他の国における商標です。

Microsoft、Windows、Windows NT、Windows 2003、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。

他の会社名、製品名およびサービス名等は、それぞれ各社の商標または登録商標です。

参考文献

以下の IBM 関連資料では、ここに記載されているトピックについて説明しています。資料は、次のトピック別に大きく分けてあります。

- CS/AIX バージョン 6.3
- IBM Communications Server for AIX, バージョン 4.2
- Redbooks™
- AnyNet/2 および SNA
- ブロック・マルチプレクサーおよび S/390 ESCON チャンネル PCI アダプター
- AIX オペレーティング・システム
- システム・ネットワーク体系 (SNA)
- ホスト構成
- z/OS Communications Server
- マルチプロトコル・トランスポート・ネットワーキング
- TCP/IP プロトコル (TCP/IP)
- X.25
- 拡張プログラム間通信機能 (APPC)
- プログラミング
- その他の IBM ネットワーキング・トピック

CS/AIX 関連の資料については、簡単な説明を付記してあります。その他の資料については、タイトル、資料番号を記し、一部の資料については本書で使用している資料タイトルの省略名を記しています。

CS/AIX バージョン 6.3 の資料

CS/AIX 関連資料として次のものがあります。なお、これらの資料のソフトコピー版が CD-ROM で提供されています。CD-ROM のソフトコピーへのアクセスの方法については、「*IBM Communications Server for AIX 入門*」を参照してください。これらのソフトコピー・ブックをシステムにインストールするには、9 から 15 MB のハード・ディスク・スペースが必要になります (このスペースは、どの各国語バージョンをインストールするかによって異なります)。

- *IBM Communications Server for AIX CS/AIX 移行ガイド* (邦文番号 SC88-6949: 英文番号 SC31-8585)

この資料は、Communications Server for AIX バージョン 4.2 以前のバージョンから CS/AIX バージョン 6 への移行方法を説明しています。

- *IBM Communications Server for AIX 入門* (邦文番号 GC88-6947: 英文番号 GC31-8583)

この資料は CS/AIX の概要を示すもので、サポートされているネットワークの特性、インストール、構成、および操作について説明しています。

- *IBM Communications Server for AIX 管理ガイド* (邦文番号 SC88-6950: 英文番号 SC31-8586)

この資料では、SNA および CS/AIX の概要、および CS/AIX の構成と操作について説明しています。

- *IBM Communications Server for AIX 管理コマンド・リファレンス* (邦文番号 SD88-6675: 英文番号 SC31-8587)

この資料では、SNA および CS/AIX のコマンドについて説明しています。

- *IBM Communications Server for AIX CPI-C プログラマーズ・ガイド* (邦文番号 SC88-6954: 英文番号 SC31-8591)

この資料では、「C」または Java™ の熟練したプログラマーを対象として、CS/AIX CPI 通信 API を使用する SNA トランザクション・プログラムの作成に関する情報を提供しています。

- *IBM Communications Server for AIX APPC プログラマーズ・ガイド* (邦文番号 SC88-6953: 英文番号 SC31-8590)

この資料では、拡張プログラム間通信機能 (APPC) を使用するアプリケーション・プログラムを作成するために必要な情報を記載しています。

- *IBM Communications Server for AIX LUA プログラマーズ・ガイド* (邦文番号 SC88-6955: 英文番号 SC31-8592)

この資料では、従来型 LU アプリケーション・プログラミング・インターフェース (LUA) を使用してアプリケーション・プログラムを作成するために必要な情報を記載しています。

- *IBM Communications Server for AIX Common Service Verb プログラマーズ・ガイド* (邦文番号 SC88-6956: 英文番号 SC31-8593)

この資料では、Common Service Verbs (CSV) アプリケーション・プログラミング・インターフェース (API) を使用してアプリケーション・プログラムを作成するために必要な情報を記載しています。

- *IBM Communications Server for AIX Management Services プログラマーズ・ガイド* (邦文番号 SC88-6957: 英文番号 SC31-8594)

この資料では、Management Services (MS) API を使用してアプリケーション・プログラムを作成するために必要な情報を記載しています。

- *IBM Communications Server for AIX Node Operator Facility プログラマーズ・ガイド* (邦文番号 SC88-6958: 英文番号 SC31-8595)

この資料では、Node Operator Facility (NOF) API を使用してアプリケーション・プログラムを作成するために必要な情報を記載しています。

- *IBM Communications Server for AIX 診断用ガイド* (邦文番号 SC88-6951: 英文番号 SC31-8588)

この資料では、SNA ネットワークの問題解決について説明しています。

- *IBM Communications Server for AIX AnyNet® ガイド: APPC over TCP/IP* (邦文番号 GC88-6961: 英文番号 GC31-8598)

この資料では、CS/AIX の AnyNet APPC over TCP/IP 機能のインストール、構成、および使用法について説明しています。

- *IBM Communications Server for AIX AnyNet ガイド: Sockets over SNA* (邦文番号 GC88-6960: 英文番号 GC31-8597)

この資料では、CS/AIX の AnyNet Sockets over SNA 機能のインストール、構成、および使用法について説明しています。

- *IBM Communications Server for AIX APPC Application Suite ユーザーズ・ガイド* (邦文番号 SC88-6959: 英文番号 SC31-8596)

この資料では、CS/AIX で使用される APPC アプリケーションについて説明しています。

- *IBM Communications Server for AIX 用語集* (邦文番号 SC88-6952: 英文番号 GC31-8589)

この資料は、IBM Communications Server for AIX 関連の資料全体で使用される用語とその定義を包括的に収録しています。

IBM Communications Server for AIX バージョン 4.2 関連資料

以下に挙げた資料は、Communications Server for AIX の以前のリリースのもので、バージョン 6 には適用されません。これらの資料は、バージョン 6 には含まれていませんが、継続してサポートされている情報を参照する場合に役に立ちます。

- *IBM Communications Server for AIX Transaction Program Reference*. (英文番号 SC31-8212)

この資料では、トランザクション・プログラミング API に関するバージョン 4.2 の情報を記載しています。バージョン 4.2 の API を使用するように作成されたアプリケーションは、バージョン 6 でも使用することができます。

IBM Redbooks

IBM は、Redbooks として知られている資料を作成している International Technical Support Center を運営しています。製品の資料と同様、Redbooks は SNA テクノロジーの理論的側面と実用的側面の両方を扱っています。ただし、製品に同梱される資料に書かれている内容は、Redbooks には記載されていません。

次の資料では、CS/AIX に役に立つ情報を記載しています。

- *IBM Communications Server for AIX Version 6* (英文番号 SG24-5947)
- *IBM CS/AIX Understanding and Migrating to Version 5: Part 2 - Performance* (英文番号 SG24-2136)
- *Load Balancing for Communications Servers* (英文番号 SG24-5305)

ユーザーはワールド・ワイド・ウェブ (WWW) の <http://www.redbooks.ibm.com> から、レッドブック資料をダウンロードすることができます。

ブロック・マルチプレクサーおよび S/390 ESCON チャンネル PCI アダプター 関連資料

次の資料では、ブロック・マルチプレクサーと S/390 ESCON チャンネル PCI アダプターについての情報を記載しています。

- AIX Version 4.1 Block Multiplexer Channel Adapter: User's Guide and Service Information (英文番号 SC31-8196)
- AIX Version 4.1 Enterprise Systems Connection Adapter: User's Guide and Service Information (英文番号 SC31-8197)
- AIX Version 4.3 S/390 ESCON Channel PCI: User's Guide and Service Information (英文番号 SC23-4232)
- IBM Communications Server for AIX チャンネル接続ユーザーズ・ガイド (邦文番号 SC88-6905: 英文番号 SC31-8219)

AnyNet/2 ソケットおよび SNA 関連資料

次の資料では、AnyNet/2 ソケットおよび SNA についての情報を記載しています。

- AnyNet/2 Version 2.0: Guide to Sockets over SNA (英文番号 GV40-0376)
- AnyNet/2 Version 2.0: Guide to SNA over TCP/IP (英文番号 GV40-0375)
- AnyNet/2: Guide to Sockets over SNA Gateway Version 1.1 (英文番号 GV40-0374)
- z/OS V1R2.0 Communications Server: AnyNet Sockets over SNA (英文番号 SC31-8831)
- z/OS V1R2.0 Communications Server: AnyNet SNA over TCP/IP (英文番号 SC31-8832)

AIX オペレーティング・システム関連資料

次の資料では、AIX オペレーティング・システムについての情報を記載しています。

- AIX バージョン 5.3 システム・マネージメント・ガイド: オペレーティング・システムおよびデバイス (邦文番号 SC88-6944: 英文番号 SC23-4910)
- AIX バージョン 5.3 システム・マネージメント・コンセプト: オペレーティング・システムおよびデバイス (邦文番号 SC88-6936: 英文番号 SC23-4908)
- AIX バージョン 5.3 システム・マネージメント・ガイド: コミュニケーションおよびネットワーク (邦文番号 SC88-6943: 英文番号 SC23-4909)
- AIX バージョン 5.3 パフォーマンス・マネージメント・ガイド (邦文番号 SC88-6934: 英文番号 SC23-4905)
- AIX Version 5.3 Performance Tools Guide and Reference (英文番号 SC23-4906)
- Performance Toolbox Version 2 and 3 Guide and Reference (英文番号 SC23-2625)
- AIXlink/X.25 Version 2.1 for AIX: Guide and Reference (英文番号 SC23-2520)

システム・ネットワーク体系 (SNA) 関連資料

次の資料では、SNA ネットワークについての情報を記載しています。

- *Systems Network Architecture: Format and Protocol Reference Manual—Architecture Logic for LU Type 6.2* (英文番号 SC30-3269)
- *Systems Network Architecture: Formats* (英文番号 GA27-3136)
- *Systems Network Architecture: Guide to SNA Publications* (英文番号 GC30-3438)
- *Systems Network Architecture: Network Product Formats* (英文番号 LY43-0081)
- *Systems Network Architecture: Technical Overview* (英文番号 GC30-3073)
- *Systems Network Architecture: APPN Architecture Reference* (英文番号 SC30-3422)
- *Systems Network Architecture: Sessions between Logical Units* (英文番号 GC20-1868)
- *Systems Network Architecture: LU 6.2 Reference—Peer Protocols* (英文番号 SC31-6808)
- *Systems Network Architecture: Transaction Programmer's Reference Manual for LU Type 6.2* (英文番号 GC30-3084)
- *Systems Network Architecture: 3270 データ・ストリーム プログラマー用解説書* (邦文番号 N:GA23-0059: 英文番号 GA23-0059)
- *Networking Blueprint Executive Overview* (英文番号 GC31-7057)
- *Systems Network Architecture: Management Services Reference* (英文番号 SC30-3346)

ホスト構成関連資料

次の資料では、ホスト構成についての情報を記載しています。

- *ES/9000, ES/3090 IOCP User's Guide Volume A04* (英文番号 GC38-0097)
- *3174 Establishment Controller Installation Guide* (英文番号 GG24-3061)
- *3174 制御装置: 計画の手引き 構成サポート C リリース5* (邦文番号 N:GA27-3918: 英文番号 GA27-3918)
- *OS/390 ハードウェア構成定義ユーザズ・ガイド* (邦文番号 SC88-6630: 英文番号 SC28-1848)
- *IBM 9032 の導入計画 ESCON ディレクター* (邦文番号 N:GA23-0364: 英文番号 GA23-0364)

z/OS Communications Server 関連資料

次の資料では、z/OS Communications Server についての情報を記載しています。

- *z/OS V1R7 Communications Server: SNA ネットワーク導入の手引き* (邦文番号 SC88-8928: 英文番号 SC31-8777)
- *z/OS V1R7 Communications Server: SNA Diagnostics* (英文番号 Vol 1: GC31-6850、Vol 2: GC31-6851)
- *z/OS V1R6 Communications Server SNA リソース定義解説書* (邦文番号 SC88-8929: 英文番号 SC31-8778)

マルチプロトコル・トランスポート・ネットワーキング関連資料

次の資料では、マルチプロトコル・トランスポート・ネットワーキング・アーキテクチャーについての情報を記載しています。

- Multiprotocol Transport Networking: Formats (英文番号 GC31-7074)
- Multiprotocol Transport Networking Architecture: Technical Overview (英文番号 GC31-7073)

TCP/IP 関連資料

次の資料では、伝送制御プロトコル / インターネット・プロトコル (TCP/IP) ネットワーク・プロトコルについての情報を記載しています。

- *z/OS V1R7 Communications Server: IP 構成ガイド* (邦文番号 SC88-8926: 英文番号 SC31-8775)
- *z/OS V1R7 Communications Server: IP 構成解説書* (邦文番号 SC88-8927: 英文番号 SC31-8776)
- *z/VM V5R1 TCP/IP 計画およびカスタマイズ* (邦文番号 SD88-6453: 英文番号 SC24-6125)

X.25 関連資料

次の資料では、X.25 ネットワーク・プロトコルについての情報を記載しています。

- *AIXLink/X.25 for AIX: Guide and Reference* (英文番号 SC23-2520)
- *RS/6000® AIXLink/X.25 Cookbook* (英文番号 SG24-4475)
- *Communications Server for OS/2 Version 4 X.25 Programming* (英文番号 SC31-8150)

APPC 関連資料

次の資料では、拡張プログラム間通信機能 (APPC) についての情報を記載しています。

- *APPC Application Suite V1 User's Guide* (英文番号 SC31-6532)
- *APPC Application Suite V1 Administration* (英文番号 SC31-6533)
- *APPC Application Suite V1 Programming* (英文番号 SC31-6534)
- *APPC Application Suite V1 Online Product Library* (英文番号 SK2T-2680)
- *APPC Application Suite Licensed Program Specifications* (英文番号 GC31-6535)
- *z/OS V1R2.0 Communications Server: APPC Application Suite User's Guide* (英文番号 SC31-8809)

プログラミング関連資料

次の資料では、プログラミングについての情報を記載しています。

- *共通プログラミング・インターフェース コミュニケーション・インターフェース CPI-C 解説書* (邦文番号 SC88-7217: 英文番号 SC26-4399)

- *Communications Server for OS/2 Warp* 日本語版 32ビット アプリケーション・プログラミングの手引き (邦文番号 SC88-5585: 英文番号 SC31-8152)

その他の IBM ネットワーキング関連資料

次の資料では、CS/AIX に関連するその他のトピックについての情報を記載していません。

- *同期データ・リンク制御 (SDLC) 解説書* (邦文番号 N:GA27-3093: 英文番号 GA27-3093)
- *Local Area Network Concepts and Products: LAN Architecture* (英文番号 SG24-4753)
- *Local Area Network Concepts and Products: LAN Adapters, Hubs and ATM* (英文番号 SG24-4754)
- *Local Area Network Concepts and Products: Routers and Gateways* (英文番号 SG24-4755)
- *Local Area Network Concepts and Products: LAN Operating Systems and Management* (英文番号 SG24-4756)
- *ネットワーク制御プログラム 資源定義の手引き* (邦文番号 N:SC30-3349: 英文番号 SC30-3349)

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]

アクセシビリティ 181
ウィンドウ・ハンドル 8
エントリー・ポイント 14

[カ行]

確立、SSCP セッションの 8
関数呼び出し、LUA に対する 16
キーボード 181
急送フロー 4
共通データ構造 52, 54
コールバック・ルーチン 7, 16
構成情報 5, 44
肯定応答 42, 44
互換性、IBM OS/2 拡張版との 48
子プロセス 46
コンパイル、AIX アプリケーションの 46
コンパイル、Linux アプリケーションの 46
コンパイルとリンク 47

[サ行]

サンプル・アプリケーション
構成 173
実行 174
処理の概要 171
テスト 173
ホスト要件 173
ショートカット・キー 181
除去 42, 44
身体障害 181
セグメント化 41
1 次 RUI 43

[タ行]

他の環境への移植性 48
通常フロー 4
特定データ構造 58

[ハ行]

ペーシング 41
1 次 RUI 43

[マ行]

マルチプロセス 46
戻りコード
1 次 177
2 次 177

[ヤ行]

予約済みパラメーター 48, 51, 63, 115

[ラ行]

リンク、AIX アプリケーションの 46
リンク、Linux アプリケーションの 46
例、LUA 通信順序の 8

[数字]

1 次戻りコード 177
2 次戻りコード 177

A

ACTLU 8
AIX アプリケーション
コンパイルとリンク 46
AIX 環境に関する考慮事項 45
ASCII から EBCDIC への変換 174

B

BIND 8
BIND パラメーター、折衝 38, 39

C

CANCEL 42, 44
CSV CONVERT verb 174

E

EBCDIC から ASCII への変換 174

G

GetLuaReturnCode 呼び出し 28

I

INITSELF 8

L

Linux アプリケーション

コンパイルとリンク 46

LU セッション 3

LU タイプ 1

LU プール 5, 46

LUA verb の概要

RUI 5

SLI 6

LUA verb、発行 35

LUA エントリー・ポイント 14

Windows 16

LUA の概念 1

LUA の定義 1

LUA_VERB_RECORD データ構造 52

N

NOTIFY 8

P

PU-SSCP セッション 3

R

RU 1, 2

RUI 2

RUI verbs の概要 5

RUI エントリー・ポイント 14

RUI と SLI、比較 2

RUI_BID

指定パラメーター 63

使用法と制約事項 70

他の verb との関連 69

戻りパラメーター 64

RUI_INIT

指定パラメーター 70

使用法と制約事項 78

他の verb との関連 77

RUI_INIT_PRIMARY

指定パラメーター 78

使用法と制約事項 82

他の verb との関連 82

RUI_PURGE

指定パラメーター 83

RUI_PURGE (続き)

他の verb との関連 87

戻りパラメーター 84

RUI_READ

指定パラメーター 87

使用法と制約事項 96

他の verb との関連 95

戻りパラメーター 89

RUI_REINIT

指定パラメーター 97

使用法と制約事項 101

他の verb との関連 100

戻りパラメーター 98

RUI_TERM

指定パラメーター 101

他の verb との関連 105

戻りパラメーター 102

RUI_WRITE

指定パラメーター 106

使用法と制約事項 113

他の verb との関連 113

戻りパラメーター 108

S

SDT 8

SLI 2

SLI verbs の概要 6

SLI エントリー・ポイント 14

SLI と RUI、比較 2

SLI_BID

指定パラメーター 115

使用法と制約事項 122

他の verb との関連 122

戻りパラメーター 116

SLI_BIND_ROUTINE

指定パラメーター 165

使用法と制約事項 166

他の verb との関連 166

戻りパラメーター 165, 166

SLI_CLOSE

指定パラメーター 123

使用法と制約事項 129

他の verb との関連 128

戻りパラメーター 124

SLI_OPEN

指定パラメーター 130

使用法と制約事項 140

他の verb との関連 139

戻りパラメーター 135

SLI_PURGE

指定パラメーター 140

他の verb との関連 145

戻りパラメーター 141

SLI_RECEIVE

指定パラメーター 145

SLI_RECEIVE (続き)
 使用法と制約事項 154
 他の verb との関連 154
 戻りパラメーター 147

SLI_SDT_ROUTINE
 指定パラメーター 166
 使用法と制約事項 167
 他の verb との関連 167
 戻りパラメーター 167

SLI_SEND
 指定パラメーター 155
 使用法と制約事項 164
 他の verb との関連 163
 戻りパラメーター 157

SLI_STSN_ROUTINE
 指定パラメーター 168
 使用法と制約事項 169
 他の verb との関連 169
 戻りパラメーター 168

SNA コンポーネント、LUA 通信に使用される 3, 4

SNA 情報 38
 1 次 RUI 42

SNA センス・コード 39, 49
 値 178

SNA メッセージ、LUA verb との関係 9

SSCP 3

SSCP セッション 3

U

UNBIND 8

V

VCB
 共通データ構造 51
 形式 51
 構造体 14
 特定データ構造 51

verb の同期完了 15, 23, 34

verb の非同期完了 7, 15

W

Windows 環境に関する考慮事項 47



プログラム番号: 5765-E51

Printed in Japan

SC88-6955-02



日本アイ・ビー・エム株式会社
〒106-8711 東京都港区六本木3-2-12