Binary for Linux - INAP

Release Notes for Version 4.00

1. Overview

This release adds support for licensing of INAP for a particular machine. In addition to this a fault has been cleared regarding the internal signal queue becoming exhausted when a large number of invokes are still active when the dialogue is aborted.

This release is fully backward compatible with the previous release. This is the first release since V3.01.

2. New functionality

2.1 Support for end user licensing

Prior to this release the use of Host Protocol Software required a signed licence agreement between the supplier and the end user.

This release offers the ability to license the software for a specific target machine using a licence file obtained electronically without the need to enter into a customer specific licence agreement.

To determine whether a licence file is required, run the binary using the –v command line option. If a licence file is required please refer to the *Host Protocol Licensing User Guide*. Note that a separate licence file is required for each target machine.

3. Faults cleared

3.1 Running out of internal signals

In previous software releases, if large numbers of invokes where active in a single dialogue at the same time when the dialogue was aborted or ended then an internal signal queue could become exhausted. This would result in a failure to free the excess number of the invoke structures. The previous releases would therefore only support up-to 32 invokes per dialogue.

The exhaustion of the internal signal queue would be indicated by the module sending MGT_MSG_EVENT_IND (0x0008) message with a status value of ERR_NO_SDLSIG (0x2e hex, 46 dec). If the signal queue were low but not exhausted, then a MGT_MSG_EVENT_IND (0x0008) message with a status value of ERR_SDLSIG_LOW (0x2f hex, 47 dec) would be sent.

This new release prevents a separate internal signal being required for each invoke and therefore removes the limitations on the number of

invokes per dialogue. There is no effect on the total number of invokes that can be simultaneously supported in the module.

Dialogic
13-Feb-03

Binary for Linux - INAP

Release Notes for Version 4.01

1. Overview

This release of INAP introduces full support for CAP V3 and provides support for the User Information parameter in dialogue messages. In addition a number of faults have been rectified regarding results and application contexts.

This release is fully backward compatible with the previous release.

2. New Functionality

2.1 CAMEL Application Part V3 Development

In this release CAMEL V3 has been implemented in full, with GPRS and SMS operations included. Further information on the CAMEL V3 implementation can be found in the INAP Programmers Manual, Issue 9.

2.2 Transmission of User Information parameter

The INAP API has been modified to allow customers to format and recover the User Information parameter in dialogue messages. The parameter is shown in the table below,

Parameter name	Size	Value (dec)
INDP_user_info	1..200 octets	10

Users of ETSI CS-1, ETSI CS-2, ITU-T and CAMEL V1, V2 and V3 protocols will be able to use these new parameters.

3. Faults cleared

3.1 Incorrect Application Context

CAMEL V1 and V2 previously had part of their application context incorrectly coded. The affected application contexts are:

INCAP_V1_GSMSSF_TO_GSMSCF
INCAP_V2_GSMSSF_TO_GSMSCF
INCAP_V2_ASSIST_HANDOFF_GSMSSF_TO_GSMSCF

The application contexts have now been corrected.

3.2 Incorrect Formatting and Recovery of Results

Previously when a result is formatted a sequence and operation code would always be added even if no parameter was present. Additionally when recovering a result, it would be rejected if the result had no sequence and operation code.

The INAP API has been changed so that when formatting a result the sequence and operation code would only be added if a parameter is present. On recovery a result will decode correctly whether or not the sequence and operation code are present.

This change will affect users of ETSI CS1, ETSI CS2, ITU CS1, CAP V1, CAP V2 and CAP V3.

3.3 Incorrect Coding of ResetTimer Operation

The ResetTimer operation was incorrectly coded in the API. This would mean that outgoing messages with this operation would be encoded incorrectly. Also correctly encoded incoming messages will result in the Extensions parameter appearing inside the Ellipsis parameter. This will only affect users using the Extensions parameter in the ResetTimer operation.

This change will affect users of ETSI CS1, ITU CS1 and CAP V2.

3.4 Operation Code Retrieval for Results

The IN_get_operation() function is intended to retrieve the operation code. An omission in IN_decode_result() meant that IN_get_operation() would not return the operation code when decoding results as expected. This has been rectified and the function can be used with results.

This change will affect users of ETSI CS1, ETSI CS2, ITU CS1, CAP V2 and CAP V3.

Dialogic
26-Jun-03

Binary for Linux - INAP

Release Notes for Version 4.02

1. Overview

This INAP release introduces new functionality that will can inform the INAP user when a message can not be sent out to the network. In addition modifications have been made that will ensure that INAP will continue to process an incoming dialogue message even when some parameters may be unknown. Included also is enhanced support for extension types.

A fault has been rectified in the area of recovery of indefinite length parameters

This release is fully compatible with the previous release.

2. New Functionality

2.1 Return on error

Previously INAP had no mechanism to inform the user application that a message could not be delivered to its destination. By using the INAP QoS parameter it is possible to instruct TCAP and SCCP to return a message if there is an error. Although TCAP could indicate to INAP that a message could not be delivered, INAP did not support this message. The QoS parameter is formatted in the same way the TCAP QoS parameter is formatted. This is shown in the table following. For more information see the TCAP Programmer's Manual issue 7.

Parameter name	INAPPN_qos
Parameter length	Either 1, 2 or 3 octets.
Parameter data	The first octet is an indicator octet, which must always be present. Subsequent octets must only be present if the appropriate bit is set in the indicator octet. The coding is as follows: ***Indicator Octet*** : **bit 0** - Set to 1 if the Return Option is selected. **bit 1** - Set to 1 if Sequence Control is required. **bit 2** - Set to 1 if the SLS Key octet is present in the Quality of Service parameter, in which case it will be the following octet. Otherwise TCAP will generate the SLS key (for passing to SCCP) automatically and the SLS key octet is omitted. **bit 3** - Set to 1 if the Message Priority Octet is included in the Quality of Service Parameter. Otherwise TCAP will insert the default message priority. All other bits are reserved for future use and must be set to zero. ***SLS Key Octet:*** The SLS Key which is used (by SCCP) to determine the SLS value to be used in the resulting message. ***Message Priority Octet:*** Coded as 0, 1, 2 or 3 to indicate the required message priority.

When a message has been returned because it can not be sent to the peer for some reason the parameters shown below are used to indicate this. The INAPPN_prob_diag parameter is now able to accept an additional value (3) indicating that a message could not be sent to the peer. This value also indicates that an additional parameter, INAPPN_report_cause, will be provided indicating the problem as reported by SCCP.

Parameter name	INAPPN_prob_diag
Parameter length	Fixed, set to one octet
Parameter data	Used to indicate unexpected events that are not related to an active operation invocation. 0 – abnormal event detected by peer 1 – response rejected by peer 2 – abnormal event received from peer 3 – abnormal network report cause

Parameter name	INAPPN_report_cause
Parameter length	Fixed, set to 1 octet
Parameter data	Values as defined in Q713 Return cause

2.2 Extension types

An enhancement to the support of operation extensions has been made. The INAP API will now support the use of an ObjectIdentifier tag that may be used as an alternative to the integer tag when formatting or recovering extension types. To facilitate this, a new parameter has been created and should be used instead of the INPN_Extension_Type when the intention is to use the ObjectIdentifier tag.

Parameter	Value	Notes
INPN_Extension_Type(n)	560-575	Used when formatting extensions to include Integer tag.
INPN_Extension_Type_ObjId(n)	898-913	Used when formatting extensions to include ObjectIdentitifer tag.

2.3 Unknown dialogue parameters

INAP has been changed so that unknown parameters in incoming dialogue messages do not cause the entire message to be discarded. Only the unknown parameters will be discarded.

3. Faults cleared

3.1 Recovery of indefinite length encoding

In the previous version of INAP an issue arose when decoding messages with indefinite lengths. When a zero length parameter was encoded using indefinite lengths within a constructed type, incorrect lengths were generated. This has been rectified.

Dialogic
17-Dec-03

Binary for Linux – INAP

Release Notes for Version 4.03

1. Overview

This release introduces the ability to run the binary for up to 10 hours at a time without the requirement for a software licences for development purposes. It also increases the maximum number of simultaneous dialogues that are supported.

This release is fully backward compatible with the previous release.

2. New functionality

2.1 Increased dialogues

This release of the module increases the number of simultaneous dialogues and associated resources that can be supported by the module. The number of resources available in the old and new releases is shown below. These additional dialogues and invokes can only be used if the main configuration message (INAP_MSG_CONFIG) is changed to enable the larger number.

Resource	Old	New
Dialogues (Incoming and Outgoing)	16384	65535
Invokes	16384	65535

2.2 Trial mode

An additional command line parameter '-t' has been added. If this flag is set then the binary will run in trial mode and the presence of a valid license certificate will not be enforced. The binary will, however, terminate after ten hours.

The licensing command line parameters (-Lp, and -Lt) should not be used when the trial mode is configured.

Dialogic
17-Mar-04

Binary for Linux – INAP

Release Notes for Version 4.04

1. Overview

This release provides a new runtime option to enable INAP to notify the application of the reception of TC-CONTINUE messages which contain has no TCAP components and improved fault diagnostics by allowing additional debugging information to be traced and also corrects minor issues as described below.

This release is fully backward compatible with the previous release.

2. New functionality

2.1 INAPDT_DELIMITER_IND on Null TC-CONTINUE

A new flag, INAPF_NULL_TC_CONT (0x0008) has been added to the <option> field of INAP_MSG_CONFIG (0x77f4) message. If this flag is set, the INAP module will generate an INAPDT_DELIMITER_IND on receipt of TC-CONTINUE without any component in the established state. If the flag is not set, the Null TC-CONTINUE will be discarded (as in previous software releases).

2.2 Debugging information

A new mechanism is now supported that enables debugging information to be traced.

Two new messages, INAP_MSG_DEBUG_MASK (0x57fe) and INAP_MSG_DEBUG_IND (0x7ff) have been introduced for this purpose. These messages, whose content can only be interpreted by Intel, should only be used on request of the Dialogic support team.

3. Faults cleared

3.1 Open Request without A/C (ITU)

Prior to this release, the module rejected an INAPDT_OPEN_REQ primitive without Application Context in ITU if the transparent AC handling option was enabled in INAP_MSG_CONFIG (bit 1, INAPF_TRNS_AC). This prevented the support of operations without Application Context.

This has now been corrected, and if the INAPF_TRNS_AC option of INAP_MSG_CONFIG is enabled in ITU, an INAPDT_OPEN_REQ primitive will be accepted by the module whether or not the Application Context parameter is present.

3.2 Release of INAP operations

In previous software releases, if many invokes were used within a dialogue, INAP operations did not always time out properly if INAP was configured for less invokes than the maximum capability of the module e.g. 65535 invokes. This issue has been corrected in this release.

Dialogic
23-Aug-05

Binary for Linux – INAP

Release Notes for Version 4.05

1. Overview

This release extends the INAP module to support for multiple Network Contexts. Each Network Context allows INAP to support a different set of options. When used with the SIU mode of the Dialogic® SS7G2x Signaling Gateway and DTS functionality this module can be used to support multiple local point codes and mixed ANSI and ITU-T configurations. This will require V3.00 or later of the SS7G2x binary distribution.

This release must be used in conjunction with TCAP for Linux V5.08 or later in order to support Network Context handling.

This release is fully backwards compatible with the previous release. Customers not using this module in conjunction with the SIU, TCAP and DTS need not upgrade.

2. New functionality

2.1 Network Context Configuration

A new configuration message INAP_MSG_NC_CONFIG (0x77f8) has been added to allow the configuration of additional Network Contexts. The INAP_MSG_CONFIG (0x77f0) message should be used to configure the default Network Context for the first network. For each subsequent Network Context the message INAP_MSG_NC_CONFIG is required. The INAP_MSG_NC_CONFIG message contains parameters to define address format and INAP specific options and therefore allows different behaviour for the module to be selected depending on the Network Context of the dialog. Section 2.2 covers how this is selected using the INAPPN_NC parameter.

The meaning of the parameters in the INAP_MSG_NC_CONFIG message is the same as the equivalent parameters in the INAP_MSG_CONFIG message. When used to support multiple local point codes within the same network the options settings should typically be the same in both messages.

INAP Message INAP_MSG_NC_CONFIG

Synopsis:

Message used to configure the additional Network Context TCAP module for operation.

Message Format:

Additional of NC specific options via INAP_MSG_NC_CONFIG message.

MESSAGE HEADER	
FIELD NAME	**MEANING**
type	**INAP_MSG_NC_CONFIG** (0x77f8)
id	Network Context id (value 1 to 3)
src	Sending module_id
dst	**INAP_TASK_ID**
rsp_req	used to request a confirmation
hclass	0
status	0
err_info	0
len	40

PARAMETER AREA		
OFFSET	**SIZE**	**NAME**
0	1	**Cnf ver**
1	1	**user_id**
2	1	**TCAP_id**
3	2	**Options**
5	35	reserved

Network Context id

The Network Context id will identify the Network Context being defined. The default Network Context (0) is configured using the existing INAP_MSG_CONFIG message therefore this message should only be used for Network Contexts 1 to 3. This assumes that four Network Contexts are permitted.

Note: All other Parameters have the equivalent meaning and types as those used in the INAP_MSG_CONFIG.

2.2 Network Context Message Handling

When a dialog is initiated remotely no change is required as INAP will automatically determine which Network Context is appropriate. An indication of the Network Context for the dialog will be passed up to the

user in the INAPPN_NC parameter. Where the dialog is initiated locally then the application needs to specify which Network Context the message is destined for. This also indicates which point code will be used as the originating point code.

The Network Context should be indicated in the first message for the dialog being used. In the case of INAP this will be in INAP Open Request using the INAPPN_NC parameter.

If no Network Context is specified then the default Network Context NC0 is assumed.

Parameter name	INAPPN_ NC (31)
Parameter length	Variable, typically 1. Length of zero indicates Network Context is unknown.
Parameter data	Network Context Identifier If the default NC is being used then this parameter is optional. If present it should have a value of 0. For other Network Contexts it should match the value defined in the relevant INAP_MSG_NC_CONFIG message.

Binary for Linux – INAP

Release Notes for Version 5.00

1. Overview

This release expands the INAP module and the INAP API to support Long Messages for the support of SCCP Segmentation and Reassembly procedures.

This is the first release since V4.05. This release is developed for use with the SS7 Development Package for Linux V5.00 or later. It cannot be used with earlier development packages.

Customers who wish to make use of the Long Message support offered in V5.00 of the development package should upgrade to this release of software (Long Message support is required for SCCP Segmentation). In addition the customer should upgrade to SCCP for Linux V3.00 or later and TCAP for Linux V6.00 or later. Other customers need not upgrade.

The INAP API is now delivered as a Linux shared object, giving the user the potential in the future to upgrade the library without needing to re-link the application process.

When using the new INAP API shared object, it is necessary to compile with the IN_LMSGS #define to get the correct function prototypes described in section 2.3 below.

2. New functionality

2.1 Support for Long Messages in INAP module

Starting with this release the INAP module now supports the generation and reception of long TCPPN_COMPONENT (towards and from the TCAP) and long INAPPN_COMPONENT (towards and from the application). This is typically used to support SCCP Segmentation and Reassembly procedures together with the SCCP and TCAP modules.

To enable correct operation the INAPF_SEGMENTATION (0x0010) flag of the option field of INAP_MSG_CONFIG (0x77f4) must be set and the GCT environment must be set up to handle long messages.

See ga237sss.pdf for more details on the new encoding scheme allowing supporting parameters longer than 255 bytes in the INAP to TCAP interface.

2.2 Use of Linux shared object

This release must be used with the shared object version of the GCT library included in V5.00 of the development package. As well as making use of the GCT shared object, the INAP API itself is also available as a shared object and this section details the installation instructions for this.

Note: Typically these steps need to be performed by a user with root privileges.

These step assume the installation instructions for the development package have already been followed and the GCT library is located in /opt/dpklnx.

Copy the INAP API library shared object to the same directory as the GCT library (e.g. /opt/dpklnx).

```
cp libin_api.so.1.0.0 /opt/dpklnx
```

If it has not already been done then file /etc/ld.so.conf should be edited and a line added indicating the path to the shared object e.g. /opt/dpklnx.

Then ldconfig should be run to configure the dynamic linker's run time configuration.

```
ldconfig -v
```

A series of links will be configured including one similar to the following.

```
/opt/dpklnx:
     libgctlib.so.1 -> libgctlib.so.1.0.0
     libin_api.so.1 -> libin_api.so.1.0.0
```

Re-linking existing applications

Any customer applications that make use of the GCT environment must be re-linked to make use of the new library, `libin_api.so.1.0.0`, and the new gctlib library in the makefile instead.

In the install directory create the following softlink:

```
ln -s libin_api.so.1 libin_api.so
```

If the new shared object version of the INAP API library is used then the linker option which links in the library should be changed. For use with the INAP API library file called `libin_api.so` , the option `in_api` becomes `–lin_api`. For example the linker command line in the makefile might appear similar to the following:

```
$(LINKER) –o$(BINARY) $(OBJS) -L/opt/dpklnx –
lin_api
```

Refer to the linker documentation for further details.

2.3 Support for Long Messages in INAPAPI

INAP API changes

To enable the INAP API library to effectively make use of longer parameters and messages it was necessary for a new version of the library to be created with a slightly modified interface.

The new INAP API library is also implemented as a shared object in order to allow applications to make use of updated versions of the library without recompilation. For backwards compatibility, the old INAPAPI is still available as a statically linked library without support for Long Messages.

Functions Prototypes

If compiled with the IN_LMSGS #define the following prototypes are used. These are contained in the file in_inc.h

int IN_init_component(void *prot_spec, IN_CPT *cpt, u32 options);

HDR *IN_alloc_message(u32 options);

int IN_set_dialogue_param(u16 param, u16 len, u8 *dptr, HDR *h);

int IN_get_dialogue_param(u16 param, u16 *dlen, u8 *dptr, u16 max_len, HDR *h);

int IN_set_component_param (u16 param_name, u16 len, u8 *data_ptr, IN_CPT *cpt);

int IN_set_param_length_range(void *prot_spec, u16 param_name, u16 length_min, u16 length_max);

int IN_get_component_param(u16 param_name, u16 *data_len, u8 *data_ptr, u16 max_len, IN_CPT *cpt);

Function	Change	Notes
IN_init_component	Addition of options field	Bit 0 (IN_INIT_OPTION_CODE_SHIFT) set to indicate if Code Shift is supported (may be generated to INAP)
IN_alloc_message	Addition of options field	Bit 0 (IN_ALLOC_OPTION_LMSGS) set to indicate if the allocated message is a large message or not.
IN_set_dialogue_param		Parameter len now u16
IN_get_dialogue_param		Parameter dlen now pointer to u16 Parameter max_len now u16
IN_set_component_param		Parameter len now u16
IN_set_param_length_range		Parameters length_min, length_max now u16
IN_get_component_param		Parameter data_len now pointer to u16 Parameter max_len now u16

IN_CPT Structure Changes

It is recommended that no application modify the internals of the structure. If an application does so it should be noted that the IN_CPT structure itself, as well as the databuf structure in IN_CPT have been modified to support a longer parameter area.

Additional error report in INAP_MSG_CONFIG confirmation

In order to simplify the identification of configuration errors INAP_MSG_CONFIG (0x77f4) now supports one extra field, error_offset (INAPMO_CONFIG_error_offset=24).

The module ignores this field on reception of the INAP_MSG_CONFIG message but sets the field in the INAP_MSG_CONFIG confirmation (0x37f4) if an error in the configuration is found. In this situation then the status field will also be set to a non-zero value.

The error offset field gives information about the byte offset of the parameter area in INAP_MSG_CONFIG which causes the configuration to fail. In order to do this the INAP_MSG_CONFIG message length must be long enough, i.e. at least 26 bytes. There is no change of operation when INAP_MSG_CONFIG length is less than 26 bytes

3. Other Changes

Ellipsis recovery

In previous software release, an ellipsis could only recover one “unknown” parameter: if an operation contained more than one “unknown” parameter, this may have resulted in a corrupted ellipsis.

In this software release, if more than one “unknown” parameter is present in an operation they are concatenated together in the order they appear in the message and stored in the same ellipsis parameter.

Message formatting to TCAP

In previous software release, if messages sent to INAP resulted in a TCAP component larger than 255 bytes, this may have caused an unexpected behaviour. This issue has been fixed. In this software release, an error is returned if the encoded TCAP component is larger than 255 bytes and the INAPF_SEGMENTATION option is not enabled. If the INAPF_SEGMENTATION option is set, an error is returned when the TCAP component is larger than 3000 bytes.

Dialogic
10-Mar-06
Revised 10-Jul-06

Binary for Linux – INAP

Release Notes for Version 5.01

1. Overview

This routine release updates corrects a number of faults within the INAP module.

This release is backwards compatible with the previous release.

2. New functionality

2.1 Invalid Network Context Reporting

Previous releases would send the maintenance indication INAP_MSG_MAINT_IND (0x07f9) with the following reason in the case of an invalid context being received by the module:

Mnemonic	Value	id	Details
INAPME_INVALID_NC	2	nc	Message received with unrecognized Network Context

This is now replaced with the Software Event code below which is sent as part of the error indication INAP_MSG_ERROR_IND (0x07f8):

Mnemonic	Value	id	Details
INAPSWE_INVALID_NC	11	nc	Message received with unrecognized Network Context

3. Other changes

3.1 Range Check for Dialogues

Previously if base incoming or outgoing dialogue ids are not set to 0, and the number of dialogue ids configured is set to the maximum (for instance base_i=0x1000, base_o=0x9000, n_i=0x8000 and n_o=0x8000), this resulted in an error which was not handled by the INAP module.

An improved range check has now been implemented to correct this error.

3.2 Unexpected Linked Operation

The error code generated for an unexpected linked operation was incorrect. This is now fixed and will return the following now as part of a provider error:

Parameter name	INAPPN_provider_error
Parameter length	Fixed, set to one octet
Parameter data	6: Unexpected Linked Operation - The remote system was not expecting the linked operation invocation

3.3 INAP sends REJECT to TCAP after receiving END

If the module detected a syntax error in a component indication from TCAP, it would send a REJECT even if an END had been received. This would cause TCAP to assume that this REJECT belongs to another dialogue causing the abort of valid dialogues.

This issue has now been resolved and will not send an REJECT if an END has been received.

3.4 User AC not supported is not passed to TCAP

When the INAP module has the Transparent AC option flag set (INAPF_TRNS_AC) in the INAP configuration message INAP_MSG_CONFIG (0x77f4) to the module, open indications for all Application Contexts are passed to the user. The INAP user can abort the dialog and may give a user reason of 6 (documented as 'AC not supported'). When the module generates a TCAP U_ABORT message, the Abort Reason field was incorrectly set to 2 (Other Reason) and not 1 for 'AC not supported' as expected.

This fault has now been corrected and will return the correct reason.

Dialogic
03-Jul-08