Event Replicator for Adabas

Application Programmer's Reference

Version 4.1.1

October 2022

ADABAS & NATURAL

This document applies to Event Replicator for Adabas Version 4.1.1 and all subsequent releases.

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

Copyright © 2022 Software AG, Darmstadt, Germany and/or Software AG USA, Inc., Reston, VA, USA, and/or its subsidiaries and/or its affiliates and/or their licensors.

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

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

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

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

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with Software AG.

Document ID: ARF-PROGRAMMING-411-20221108

Table of Contents

Preface v
1 About this Documentation 1
    Document Conventions 2
    Online Information and Support 2
    Data Protection 3
2 Output Message Summary 5
3 Sample Output Message 9
4 Output Message Detail 17
    URBC 18
    URBD 19
    URBE 20
    URBH 21
    URBI 22
    URBL 25
    URBP 37
    URBQ 39
    URBR 40
    URBS 41
    URBT 46
    URBU 48
    URBX 49
    URBZ 51
5 Event Replicator Client Requests 55
    Specifying the URBH 56
    Destination Open and Close Requests 57
    Initial-State Requests 60
    Prior-Transaction Requests 76
    Status Requests 80
6 Testing webMethods EntireX and WebSphere MQ Queues 85
    Using the ADAMTR (Message Test Receive) Utility 86
    Using the ADAMTS (Message Test Send) Utility 92
Index 99

Preface

The Event Replicator Server sends replication data to a target application through a messaging system. An output message sent by the Event Replicator for Adabas contains both control information (for example, transaction and record information) and payload data (for example, the replicated data).

The following DSECTs describe the types of control information:

- **URBC - Continuation element**
- **URBD - Data element**
- **URBE - End-of-transaction element**
- **URBH - Message header**
- **URBI - Input element**
- **URBL - Transaction log record element**
- **URBP - Subscription user exit parameter list element**
- **URBQ - User exit program parameter block**
- **URBR - Record element**
- **URBS - Status/response element**
- **URBT - Transaction element**
- **URBU - ADARPE extract header user element**
- **URBX - Subscription user exit parameter block element**
- **URBZ - User exit program parameter block**

For detailed information on each DSECT, see *Output Message Detail* section in this guide.

This document is organized as follows:

Output Message Summary	Provides a summary of the output messaging process.
Sample Output Message	Provides an example of the output messages that the Event Replicator will produce, in textual form.
Output Message Detail	Provides details for the URB DSECTs pertinent to Event Replicator for Adabas.
Event Replicator Client Requests	Describes Event Replicator client requests.
Testing webMethods EntireX Broker and WebSphere MQ Queues	Describes the ADAMTR and ADAMTS utilities, used to test webMethods EntireX Broker and WebSphere MQ queues without having (or prior to writing) a target application.

1 About this Documentation

- Document Conventions 2
- Online Information and Support 2
- Data Protection 3

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
`Monospace font`	Identifies service names and locations in the format *`folder.subfolder.service`*, APIs, Java classes, methods, properties.
`Italic`	Identifies: Variables for which you must supply values specific to your own situation or environment. New terms the first time they occur in the text. References to other documentation sources.
`Monospace font`	Identifies: Text you must type in. Messages displayed by the system. Program code.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.
|	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the | symbol.
[ ]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [ ] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

Online Information and Support

Product Documentation

You can find the product documentation on our documentation website at **https://documentation.softwareag.com**.

In addition, you can also access the cloud product documentation via **https://www.softwareag.cloud**. Navigate to the desired product and then, depending on your solution, go to “Developer Center”, “User Center” or “Documentation”.

Product Training

You can find helpful product training material on our Learning Portal at **https://knowledge.softwareag.com**.

Tech Community

You can collaborate with Software AG experts on our Tech Community website at **https://tech-community.softwareag.com**. From here you can, for example:

- Browse through our vast knowledge base.
- Ask questions and find answers in our discussion forums.
- Get the latest Software AG news and announcements.
- Explore our communities.
- Go to our public GitHub and Docker repositories at **https://github.com/softwareag** and **https://hub.docker.com/publishers/softwareag** and discover additional Software AG resources.

Product Support

Support for Software AG products is provided to licensed customers via our Empower Portal at **https://empower.softwareag.com**. Many services on this portal require that you have an account. If you do not yet have one, you can request it at **https://empower.softwareag.com/register**. Once you have an account, you can, for example:

- Download products, updates and fixes.
- Search the Knowledge Center for technical information and tips.
- Subscribe to early warnings and critical alerts.
- Open and update support incidents.
- Add product feature requests.

Data Protection

Software AG products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR). Where applicable, appropriate steps are documented in the respective administration documentation.

2 Output Message Summary

Following is a summary of output messaging process:

1. The output message will start with a URBH DSECT. The URBH:
   a. Identifies the message as one sent by the Event Replicator for Adabas (URBHEYE).
   b. Identifies the version of DSECTs sent in the message (URBHVERS).
   c. Identifies the Event Replicator Server sending the message (URBHRPID and URBHRPNI).
   d. Gives the length of the entire message (URBHLENT).
   e. Includes a message sequence number by destination (URBHMSNR). This value identifies the sequence of messages sent to a destination.
   f. Indicates the time that the Event Replicator Server passed the message to the messaging system (URBHTIME).
   g. Indicates whether or not more data for the transaction will follow in the next message (URBTCONT).
2. A URBT identifies the start of a transaction's worth of replicated data. The URBT:
   a. Identifies the database in which the transaction was executed (URBTDBID).
   b. Identifies the related subscription (URBTSNAM).
   c. Identifies the transaction sequence number within a subscription (URBTTSNR).
   d. Identifies the number of records (URBRs) in the transaction (URBTRCNT).
   e. Indicates the time that Adabas processed the ET commit (URBTTIM).
   f. Indicates the time that the Event Replicator Server completed subscription processing for the transaction (URBTPTIM)
3. A URBR identifies a record within a transaction. The URBR:
   a. Identifies the record that was updated (URBRFNR and URBRISN).
   b. Identifies the record sequence number within a transaction (URBRRSNR).

   c. Identifies the number of data elements (URBDs) related to this record (URBRDCNT).

4. A URBD identifies and contains a before or after image of replicated data for one record. The URBD:
   a. Identifies the data element sequence number for within a record (URBDDSNR).
   b. Identifies the type of data (URBDTYP).
5. A URBE identifies the end of a transaction's worth of replicated data. Note: The end of a transaction's worth of replicated data may also be identified by:
   a. Determining that the number of records (URBRs) received is equal to the number of records (URBTRCNT) in the transaction, and
   b. Determining that all data elements (URBDs) have been received for each record (URBRDCNT).
6. A URBT will be sent prior to a URBR for the same transaction.
7. A URBR will be sent prior to a URBD for the same record.
8. A URBR will NOT always be followed by a URBD for the record. For example, if an error occurs that prevents decompression of the after image and there is no before image, a URBR will be sent with no URBD.
9. The replicated data for one transaction may be sent in one or more messages.
10. When the output message contains subsequent data for a transaction (i.e. the related URBT was in a previous message), the output message will start with a URBH followed by a URBC.
11. Replicated data for different transactions may be sent in the same message.
12. Replicated data for different transactions will NOT be interleaved. That is, all of the data for a transaction will be sent before any of the data is sent for a different transaction.
13. Replicated data for different subscriptions will not be sent in the same message.
14. A control message (URBS) will be sent during initialization of an Event Replicator Server.
15. A control message (URBS) will be sent during normal termination of an Event Replicator Server.
16. Sequence numbering (message sequence number, transaction sequence number) restarts at one after the restart of an Event Replicator Server.
17. A second or subsequent transaction's worth of data will only be put in the output message if the entire data related to that transaction fits in the remaining portion of the output message.

About Output Message Time Stamps

All timestamps given (URBHTIME, URBTTTIM, etc.) are in STCK format. The timestamps represent the machine time. Depending on the time zone settings in the operating system, this may or may not be the local time where the machine is located.

Note: See IBM's *z/Architecture Principles of Operation* (SA22-7832-02), Chapter 4, Section "Time-of-Day Clock", and Chapter 7, "STORE CLOCK" for information on the STCK format.

At the time of this documentation's publication, this IBM manual can be found at ***http://publibfi.boulder.ibm.com/epubs/pdf/dz9zr009.pdf***.

The timestamps for the last update of a data record (URBRTIME) and the transaction commit (URBTTTIM) are generated by Adabas. The timestamps for the end of subscription processing for a transaction (URBTPTIM) and the send of a message (URBHTIME) are generated by the Event Replicator Server. If Adabas and the Event Replicator Server reside on different machines, then it depends on the configuration of these machines whether these timestamps are comparable or by how much they may out of sync.

Provided these timestamps are comparable, they can be used to determine at which points during replication processing delays occurred: between the commit of a transaction in Adabas and the subscription processing in the Event Replicator Server, or between the subscription processing and the sending of the message, or between the sending of the message and its arrival at the target application.

3 Sample Output Message

This is an example of the output messages that the Event Replicator Server will produce, in textual form. The URB* elements are attached together in top-down sequence.

Notes:

1. Message sequence number 1 is the first message sent to each destination.
2. Message sequence number 6 in the following example is the final message sent to each destination when the Event Replicator address space is terminated normally.

```
Message boundary:

URBH -- URBHMSNR = 1           Message sequence number for this
                                destination

URBS -- URBSST   = 'STRT'      Reptor status information
        URBSTIME =             Timestamp when URBS was created

Message boundary:

URBH -- URBHMSNR = 2           Message sequence number for this
                                destination

URBT -- URBTSNAM = 'SUBS1'     Subscription name
        URBTTSNR = 1           Transaction sequence number for this
                                subscription
        URBTRCNT = 4           Record count in transaction

URBR -- URBRRSNR = 1           Record sequence number for this
                                transaction
        URBRDCNT = 1           Data element count for record
        URBRTYP  = 'I'         Insert

URBD -- URBDDSNR = 1           Data element sequence number
```

```
                                 for record
        URBDTYP  = 'A'          After image of record
        URBDDATA = ...          Record data

URBR -- URBRRSNR = 2            Record sequence number for this
                                 transaction
        URBRDCNT = 2            Data element count for record
        URBRTYP  = 'U'          Update

URBD -- URBDDSNR = 1            Data element sequence number
                                           for record
        URBDTYP  = 'B'          Before image of record
        URBDDATA = ...          Record data

URBD -- URBDDSNR = 2            Data element sequence number
                                 For record
        URBDTYP  = 'A'          After image of record
        URBDDATA = ...          Record data

URBR -- URBRRSNR = 3            Record sequence number for this
                                 transaction
        URBRDCNT = 1            Data element count for record
        URBRTYP  = 'D'          Delete

URBD -- URBDDSNR = 1            Data element sequence number
                                 for record
        URBDTYP  = 'K'          Before image of primary key
        URBDDATA = ...          Key data

URBR -- URBRRSNR = 4            Record sequence number for this
                                 transaction
        URBRDCNT = 2            Data element count for record
        URBRTYP  = 'U'          Update

URBD -- URBDDSNR = 1            Data element sequence number
                                 for record
        URBDTYP  = 'B'          Before image of record
        URBDDATA = ...          Record data

URBD -- URBDDSNR = 2            Data element sequence number
                                 for record
        URBDTYP  = 'A'          After image of record
        URBDDATA = ...          Record data

URBE -- URBESNAM = 'SUBS1'      Subscription name
        URBETSNR = 1            Transaction sequence number for
                                 this subscription

URBT -- URBTSNAM = 'SUBS1'      Subscription name
        URBTTSNR = 2            Transaction sequence number for
                                 this subscription
        URBTRCNT = 1            Record count in transaction
```

```
URBR -- URBRRSNR = 1           Record sequence number for
                                this transaction
        URBRDCNT = 1           Data element count for record
        URBRTYP  = 'I'         Insert

URBD -- URBDDSNR = 1           Data element sequence number
                                for record
        URBDTYP  = 'A'         After image of record
        URBDDATA = ...         Record data

URBE -- URBESNAM = 'SUBS1'     Subscription name
        URBETSNR = 2           Transaction sequence number for
                                this subscription

URBT -- URBTSNAM = 'SUBS1'     Subscription name
        URBTTSNR = 3          Transaction sequence number for
                                   this subscription
        URBTRCNT = 1           Record count in transaction

URBR -- URBRRSNR = 1           Record sequence number for this
                                transaction
        URBRDCNT = 1           Data element count for record
        URBRTYP  = 'D'         Delete

URBD -- URBDDSNR = 1           Data element sequence number
                                for record
        URBDTYP  = 'B'         Before image of record
        URBDDATA = ...         Record data

URBE -- URBESNAM = 'SUBS1'     Subscription name
        URBETSNR = 3           Transaction sequence number for
                                this subscription


Message boundary:

URBH -- URBHMSNR = 3           Message sequence number for this
                                destination

URBT -- URBTSNAM = 'SUBS2'     Subscription name
        URBTTSNR = 1           Transaction sequence number for
                                   this subscription
        URBTRCNT = 7           Record count in transaction

URBR -- URBRRSNR = 1           Record sequence number for this
                                transaction
        URBRDCNT = 1           Data element count for record
        URBRTYP  = 'D'         Delete

URBD -- URBDDSNR = 1           Data element sequence number
                                for record
```

```
        URBDTYP  = 'B'          Before image of record
        URBDDATA = ...          Record data


URBR -- URBRRSNR = 2            Record sequence number for
                                 this transaction
        URBRDCNT = 1            Data element count for record
        URBRTYP  = 'I'          Insert

URBD -- URBDDSNR = 1            Data element sequence number
                                 for record
        URBDTYP  = 'A'          After image of record
        URBDDATA = ...          Record data

URBR -- URBRRSNR = 3            Record sequence number for
                                 this transaction
        URBRDCNT = 1            Data element count for record
        URBRTYP  = 'I'          Insert

URBD -- URBDDSNR = 1            Data element sequence number
                                 for record
        URBDTYP  = 'A'          After image of record
        URBDDATA = ...          Record data

URBR -- URBRRSNR = 4            Record sequence number for
                                 this transaction
        URBRDCNT = 1            Data element count for record
        URBRTYP  = 'I'          Insert

URBD -- URBDDSNR = 1            Data element sequence number
                                 for record
        URBDTYP  = 'A'          After image of record
        URBDDATA = ...          Record data

URBR -- URBRRSNR = 5            Record sequence number for
                                 this transaction
        URBRDCNT = 2            Data element count for record
        URBRTYP  = 'U'          Insert

URBD -- URBDDSNR = 1            Data element sequence number
                                 for record
        URBDTYP  = 'B'          Before image of record
        URBDDATA = ...          Record data

URBD -- URBDDSNR = 2            Data element sequence number
                                 for record
        URBDTYP  = 'A'          After image of record
        URBDDATA = ...          Record data

URBR -- URBRRSNR = 6            Record sequence number for
                                 this transaction
        URBRDCNT = 1            Data element count for record
```

```
        URBRTYP  = 'I'         Insert

URBD -- URBDDSNR = 1           Data element sequence number
                                for record
        URBDTYP  = 'A'         After image of record
        URBDDATA = ...         Record data


URBR -- URBRRSNR = 7           Record sequence number for
                                this transaction
        URBRDCNT = 7           Data element count for record
        URBRTYP  = 'U'         Update

URBD -- URBDDSNR = 1           Data element sequence number
                                for record
        URBDTYP  = 'B'         Before image of record
        URBDDATA = ...         Record data


Message boundary:

URBH -- URBHMSNR = 4           Message sequence number for this
                                destination

URBC -- URBCSNAM = 'SUBS2'     Subscription name
        URBCTSNR = 1           Current transaction sequence number
        URBCRSNR = 7           Current record sequence number
        URBCDSNR = 1           Current data sequence number

URBD -- URBDDSNR = 2           Data element sequence number
                                for record
        URBDTYP  = 'A'         After image of record
        URBDDATA = ...         Record data

URBE -- URBESNAM = 'SUBS2'     Subscription name
        URBETSNR = 1          Transaction sequence number for
                                this subscription

URBT -- URBTSNAM = 'SUBS2'     Subscription name
        URBTTSNR = 2           Transaction sequence number for this
                                subscription
        URBTRCNT = 2           Record count in transaction

URBR -- URBRRSNR = 1           Record sequence number for this
                                transaction
        URBRDCNT = 1           Data element count for record
        URBRTYP  = 'I'         Insert

URBD -- URBDDSNR = 1           Data element sequence number
                                for record
        URBDTYP  = 'A'         After image of record
        URBDDATA = ...         Record data
```

```
URBR -- URBRRSNR = 2           Record sequence number for this
                                transaction
        URBRDCNT = 2           Data element count for record
        URBRTYP  = 'U'         Update

URBD -- URBDDSNR = 1           Data element sequence number
                                          for record
        URBDTYP  = 'B'         Before image of record
        URBDDATA = ...         Record data

URBD -- URBDDSNR = 2           Data element sequence number
                                For record
        URBDTYP  = 'A'         After image of record
        URBDDATA = ...         Record data

URBE -- URBESNAM = 'SUBS2'     Subscription name
        URBETSNR = 2          Transaction sequence number for
                                this subscription


Message boundary:

URBH -- URBHMSNR = 5           Message sequence number for this
                                destination

URBT -- URBTSNAM = 'SUBS1'     Subscription name
        URBTTSNR = 4           Transaction sequence number for
                                this subscription
        URBTRCNT = 1           Record count in transaction

URBR -- URBRRSNR = 1           Record sequence number for
                                this transaction
        URBRDCNT = 1           Data element count for record
        URBRTYP  = 'I'         Insert

URBD -- URBDDSNR = 1           Data element sequence number
                                for record
        URBDTYP  = 'A'         After image of record
        URBDDATA = ...         Record data

URBE -- URBESNAM = 'SUBS1'     Subscription name
        URBETSNR = 4           Transaction sequence number for
                                this subscription


Message boundary:

URBH -- URBHMSNR = 6           Message sequence number for this
                                destination
```

```
URBS -- URBSST   = 'TERM'      Reptor status information
        URBSTIME =             Timestamp when URBS was created
```

4 Output Message Detail

- URBC 18
- URBD 19
- URBE 20
- URBH 21
- URBI 22
- URBL 25
- URBP 37
- URBQ 39
- URBR 40
- URBS 41
- URBT 46
- URBU 48
- URBX 49
- URBZ 51

DSECTs defining data structures known to the outside start with the characters 'URB'.

The data structures are shown in the form of an abridged assembly listing output to better fit into the documentation format. The columns have the following meaning.

Column	Meaning
Offset	The hexadecimal displacement from the beginning of the structure.
Num2	The optional hexadecimal value of literal definitions (EQU) – rightmost 2 bytes.
Num3	The optional hexadecimal size of the structure (DSECT).
Source	The assembler source line in the format: *label instruction parameters comments ↩*

Each item is separated by space. For a detailed account of the Assembler Language please consult the HLASM Manual. For example:

```
Offset Num2 Num3  Source
00018             URBCDSNR DS    F             Current data sequence number for ↩
record
```

This chapter provides details for the URB DSECTs.

URBC

```
                          URBC     ,               Replication buffer - Continuation
00000 00000 00030 URBC     DSECT
                  ↩
*-------------------------------------------------------------------*
                  *        URBC -- Continuation element                      ↩
      *
                  *                                                          ↩
      *
                  *        DSECT URBC is used by the Adabas Replication Facility. ↩
      *
                  *                                                          ↩
      *
                 *        DSECT URBC contains information related to a transaction ↩
    *
                  *        that is continued in the current message.         ↩
      *
                  *                                                          ↩
      *
                   ↩
*-------------------------------------------------------------------*
00000             URBCEYE  DS    CL4           URBC eye-catcher 'URBC'
```

```
00004             URBCLEN  DS    F              Length of URBC
                  *
00008             URBCSNAM DS    CL8            Subscription name
00010             URBCTSNR DS    F             Current transaction sequence number
                  *                             within subscription/destination
00014             URBCRSNR DS    F             Current record sequence number
                  *                             within transaction
00018             URBCDSNR DS    F             Current data sequence number for ↩
record
                  *
0001C             URBCCONT DS    C             Indicator for transaction ↩
continuation:
      00E8        URBCCONY EQU   C'Y'           More data to follow in the next ↩
message
                  *                              for the same transaction
                  *
0001D                      DS    XL19          Reserved area
00030                      DS    0D
      0030        URBCL    EQU   *-URBC         DSECT length
```

URBD

```
                           URBD     ,                Replication buffer - Data
00000 00000 00020 URBD     DSECT
                  ↩
*------------------------------------------------------------------------*
                  *        URBD -- Data element                           ↩
      *
                  *                                                       ↩
      *
                  *        DSECT URBD is used by the Event Replicator for Adabas. ↩
      *
                  *                                                       ↩
      *
                  *        DSECT URBD contains data related to a before or after  ↩
      *
                  *        image associated with one record.              ↩
      *
                  *                                                       ↩
      *
                  ↩
*------------------------------------------------------------------------*
00000             URBDEYE  DS    CL4            URBD eye-catcher 'URBD'
00004             URBDLEN  DS    F            Length of URBD + related data + filler
00008             URBDLENH DS    F              Length of URBD
0000C             URBDLEND DS    F            Length of related data (at URBDDATA)
                  *
00010             URBDDSNR DS    F              Data sequence number for record
                  *
```

```
00014             URBDTYP  DS    C               Type of data:
      00C1        URBDTYPA EQU   C'A'             After  image of data storage
      00C2        URBDTYPB EQU   C'B'             Before image of data storage
      00D2        URBDTYPK EQU   C'K'             Before image of primary key
                  *
00015                      DS    XL11            Reserved area
                  *
00020            URBDDATA DS    0D             Payload data (before or after image)
      0020        URBDL    EQU   *-URBD          DSECT length
```

URBE

```
                           URBE     ,                Replication buffer - End
00000 00000 00020 URBE     DSECT
                  ↩
*------------------------------------------------------------------------*
                  *        URBE -- End-of-transaction element                  ↩
      *
                  *                                                            ↩
      *
                  *        DSECT URBE is used by the Adabas Replication Facility. ↩
      *
                  *                                                            ↩
      *
                 *        DSECT URBE indicates the end of the transaction related ↩
     *
                  *        to the preceding record and data elements.          ↩
      *
                  *                                                            ↩
      *
                  ↩
*------------------------------------------------------------------------*
00000             URBEEYE  DS    CL4             URBE eye-catcher 'URBE'
00004             URBELEN  DS    F               Length of URBE
                  *
00008             URBESNAM DS    CL8             Subscription name
00010             URBETSNR DS    F               Transaction sequence number within
                  *                               subscription/destination
                  *
00014                      DS    XL12            Reserved area
00020                      DS    0D
      0020        URBEL    EQU   *-URBE          DSECT length
```

URBH

```
                           URBH     ,                Replication buffer - Header
00000 00000 00040 URBH     DSECT
                  ↩
*------------------------------------------------------------------------*
                  *        URBH -- Message header                         ↩
      *
                  *                                                       ↩
      *
                  *        DSECT URBH is used by the Event Replicator for Adabas. ↩
      *
                  *                                                       ↩
      *
                  *        DSECT URBH contains information related to one message ↩
      *
                  *        sent to a replication target via a messaging system    ↩
      *
                  *        (e.g., MQSeries or Entire Broker).             ↩
      *
                  *                                                       ↩
      *
                  *        DSECT URBH is put at the start of each buffer sent to  ↩
      *
                  *        the messaging system.                          ↩
      *
                  *                                                       ↩
      *
                  ↩
*------------------------------------------------------------------------*
00000              URBHEYE  DS    CL4            URBH eye-catcher 'URBH' (in EBCDIC)
00004              URBHLEN  DS    F               Length of URBH
                   *
00008              URBHVERS DS    CL2             Version indicator for URB* DSECTs:
      F0F1         URBHVER1 EQU   C'01'             First version
                   *                              Future releases may allow a new ↩
layout of
                  *                             the URB* DSECTs. In this case URBHVERS
                   *                               will be set to a different value.
0000A             URBHBORD DS    H              Constant 1 to indicate the byte order:
      0001         URBHBORH EQU   X'0001'           Big-endian (high-order byte first)
      0100        URBHBORL EQU   X'0100'          Little-endian (low-order byte first)
                   *
0000C              URBHLENT DS    F              Total message length (including URBH)
00010              URBHMSNR DS    F               Message sequence number per ↩
destination
00014              URBHTIME DS    XL8             Time (STCK) when message was sent
                   *
0001C              URBHRPID DS    H               Target ID of originating Reptor
```

```
0001E             URBHRPNI DS    H              Nucleus ID of originating Reptor
00020             URBHNAME DS    CL8            Sender name ('REPTOR' if sent by ↩
Reptor)
                  *
00028             URBHRES1 DS    XL24           Reserved area
00040                      DS    0D
      0040        URBHL    EQU   *-URBH         DSECT length
                  *
      F0F1        URBHVERC EQU   URBHVER1              Current version - Initial ↩
release
```

URBI

```
                           URBI     ,                Replication buffer - Input
00000 00000 00060 URBI     DSECT
                  ↩
*------------------------------------------------------------------------*
                  *        URBI -- Input element                          ↩
      *
                  *                                                       ↩
      *
                  *        DSECT URBI is used by the Adabas Replication Facility. ↩
      *
                  *                                                       ↩
      *
                  *        DSECT URBI contains input data for a request from a    ↩
      *
                  *        target application to the Reptor.              ↩
      *
                  *                                                       ↩
      *
                  ↩
*------------------------------------------------------------------------*
                  *
                  *  For all types of request, fill in:
                  *     *  The eye-catcher (URBIEYE).
                 *     *  The length of all data belonging to this URBI, comprising
                 *        the URBI proper, any selection data, and any filler added
                  *        for alignment or other purposes (URBILEN).
                  *     *  The length of the URBI proper (URBILENH).
                  *     *  The length of the selection data proper (URBILEND);
                  *        zero, if there is no selection data.
                *     *  An arbitrary token associated with the request (URBIRTOK).
                *        The token has no meaning to the Reptor. The Reptor returns
                 *        it in all responses to the request. The token may be used
                  *        by the requestor to associate the response(s) with the
                  *        original request.
                *     *  The destination to which the response(s) is/are to be sent
                  *        (URBIRNAM). Responses are sent to all destinations that
```

```
                *       are associated with the request. If URBIRNAM is filled in,
                *       the response(s) will additionally be sent to the specified
                  *       destination.
                  *     *  The request type (URBIRT).
                  *
                  *  Set unused fields to binary zeros or, for character fields, ↩
blanks.
                  *
                  *  Character data may be specified in either EBCDIC or ASCII. The
                  *  value provided in the eye-catcher in the URBH (URBHEYE) signals
                  *  the way character data are provided in the entire message.
                  *
                 *  Binary numbers may be specified in either big-endian (high-order
                  *  byte first) or little-endian (low-order byte first) byte order.
                 *  The value provided in the byte-order field in the URBH (URBHBORD)
                  *  signals the way binary numbers are provided in the entire ↩
message.
                  *
                  *
00000             URBIEYE  DS    CL4            URBI eye-catcher 'URBI'
00004             URBILEN  DS    F             Length of URBI + selection data + ↩
filler
00008             URBILENH DS    F              Length of URBI proper (>= URBIL)
0000C            URBILEND DS    F             Length of selection data (may be zero)
00010             URBIRTOK DS    XL8            Request ID token (returned in ↩
response)
00018             URBIRNAM DS    CL8            Destination name for response
                  *
00020             URBIRT   DS    CL4            Request type:
      C1E3        URBIRTST EQU   C'STAT'         Inquire subscription/destination ↩
status
      E2E3        URBIRTIS EQU   C'INST'          Commence initial-state processing
      C1D5       URBIRTTA EQU   C'TRAN'         Retrieve data for prior transaction
      D5C4        URBIRTOD EQU   C'OPND'          Open a destination
      E2C4        URBIRTCD EQU   C'CLSD'          Close a destination
                  *
                  *
                  *  For an initial-state request (INST), specify:
                  *     *  The initial-state name (URBIINAM).
                  *     *  The database ID (URBIDBID) and file number (URBIFNR).
                  *     *  If an ISN list or selection criterion has been defined
                  *        for the initial-state process, the selection data
                  *        (starting at URBI + URBILENH), which is either a list
                  *        of ISNs or a value buffer.
                  *
                  *  If the initial-state definition involves multiple files,
                  *  provide one URBI for each file in one message (URBH) and
                  *  specify the same request ID token, response destination,
                  *  and initial-state name for each one.
                  *
00024             URBIDBID DS    H              Database ID (INST request)
00026             URBIFNR  DS    H              File number (INST request)
```

```
00028             URBIINAM DS    CL8            Initial-state name (INST request)
                  *
                  *
                  *  For a status request (STAT), specify:
                  *     *  The subscription name or blanks (URBISNAM).
                  *     *  The destination name or blanks (URBIDNAM).
                  *
                  *  At least one of them must be given.
                 *  If both a subscription name and destination name are given, the
                  *  status of the last transaction processed for the subscription
                  *  will be sent to the destination in a URBS element.
                  *  If only a subscription name is given, the status of the last
                 *  transaction processed for the subscription will be sent in URBS
                  *  elements to every destination defined for the subscription.
                  *  If only a destination name is given, the status of the last
                  *  transaction processed for every subscription defined for the
                 *  destination will be sent in URBS elements to that destination.
                  *
                  *
                  *  For a prior-transaction request (TRAN), specify:
                  *     *  The subscription name (URBISNAM).
                  *     *  The destination name (URBIDNAM).
                  *     *  The transaction sequence number within the
                  *        subscription/destination (URBITSNR).
                  *
                  *  The replication data from the transaction identified by the
                  *  subscription and destination names and the sequence number
                  *  will be sent to the specified destination.
                  *  If the requested data is no longer available, a URBS with
                  *  related error information will be sent to the response
                  *  destination (URBIRNAM).
                  *
00030             URBISNAM DS    CL8          Subscription name or blank (STAT/TRAN)
00038             URBIDNAM DS    CL8          Destination name or blank (STAT/TRAN ↩
req)
                  *
                  *
                 *  The following three fields should be set if the selection data
                  *  (URBIDATA) has a different encoding or architecture than the
                 *  database. The usage is like the ACODE/WCODE and ARC parameters
                  *  in the record buffer for the Adabas OP command.
                  *  These fields may be used only if the Reptor is UES enabled.
                  *  They may be used for an initial-state request with selection
                  *  criterion and must be zero for all other requests.
                  *
00040             URBIACOD DS    F            Alpha field encoding in selection data
00044              URBIWCOD DS    F             Wide field encoding in selection data
00048              URBIARC  DS    X             Architecture flags for selection data
                  *
00049              URBIRES1 DS    XL3              Reserved area 1
                  *
0004C              URBITSNR DS    F                Transaction sequence number within
```

```
                  *                                subscription/destination (TRAN ↩
request)
                  *
00050             URBIRES2 DS    XL16           Reserved area 2
                  *
00060             URBIDATA DS    0D             Optional selection data (INST ↩
request):
                  *                               Value buffer or ISN list
      0060        URBIL    EQU   *-URBI          DSECT length
```

URBL

```
                           URBL     ,              Replication buffer - TLOG record
00000 00000 00040 URBL     DSECT ,              TLOG base record
                  ↩
*-------------------------------------------------------------------*
                  *        URBL - Reptor Transaction Log Record Definition       ↩
      *
                  *                                                              ↩
      *
                  *        URBL is used by the Adabas Replication Facility.      ↩
      *
                  *                                                              ↩
      *
                  *         The URBL contains the base information for one TLOG ↩
record.  *
                  *         If URBLPTYP is nonzero, additional data follows this ↩
record  *
                *        and is mapped by the DSECT indicated by the equate value. ↩
   *
                  *                                                              ↩
      *
                  ↩
*-------------------------------------------------------------------*
00000             URBLEYE  DS    CL4            URBL eye-catcher 'URBL'
00004             URBLLEN  DS    F              Length of URBL
00008             URBLTLEN DS    F             Total TLOG record length including ↩
URBL
0000C             URBLVER  DS    CL2            Version of URBL structure:
      F0F1        URBLVER1 EQU   C'01'           First version
                  *
0000E             URBLTYPE DS    CL2            Type of TLOG Record:
                  *
                  *        Reptor processing events:
                  *
      C1C1        URBLTPRL EQU   C'AA'           Lowest processing event
      C1C1        URBLTTAQ EQU   C'AA'           Transaction on assignment queue
      C1C2        URBLTNSM EQU   C'AB'           No subscription match
      C1C3        URBLTSUI EQU   C'AC'           Input to subscription processing
```

```
      C1C4        URBLTFIL EQU   C'AD'           Data record filtered
      C1C5       URBLTSUO EQU   C'AE'          Output from subscription processing
      C1C6        URBLTADS EQU   C'AF'           Assigned to destination
      C1C7        URBLTSLW EQU   C'AG'           Transaction written to SLOG
      C1C8        URBLTSLR EQU   C'AH'           Transaction read from SLOG
      C1C9        URBLTOUC EQU   C'AI'           Output completion - Message sent
      C1D1        URBLTTCQ EQU   C'AJ'           Transaction on completion queue
      C1D2        URBLTTXC EQU   C'AK'           Transaction completed - Adabas ↩
informed
      C1D3       URBLTRQR EQU   C'AL'          Request received (from application)
      C1D4        URBLTRQT EQU   C'AM'           Request translated
      C1D5        URBLTRQJ EQU   C'AN'           Request rejected (undecipherable)
      C1D6        URBLTRQE EQU   C'AO'           Request error (execution failed)
      C1D7        URBLTSTA EQU   C'AP'           Response to status request
      C1D8        URBLTRET EQU   C'AQ'           Response to retransmit request
      C1D9        URBLTISS EQU   C'AR'           Initial-state request started
      C1E2        URBLTISC EQU   C'AS'           Initial-state request completed
      C1E3        URBLTDOD EQU   C'AT'           Destination output data
      C1E4        URBLTFMA EQU   C'AU'           Filter condition matched
      C1E5        URBLTFNM EQU   C'AV'           Filter condition not matched
      C1E6        URBLTFIG EQU   C'AW'           Filter condition ignored
      C1E7        URBLTADE EQU   C'AX'           ADABAS destination error
      C2E9        URBLTPRH EQU   C'BZ'           Highest processing event
                  *
                  *        Reptor state-change events:
                  *
      D4C1        URBLTSCL EQU   C'MA'           Lowest state-change event
      D4C1        URBLTRPI EQU   C'MA'           Reptor initialized
      D4C2        URBLTRPT EQU   C'MB'           Reptor terminated (normally)
      D4C3        URBLTSBA EQU   C'MC'           Subscription activated
      D4C4        URBLTSBD EQU   C'MD'           Subscription deactivated
      D4C5        URBLTDSA EQU   C'ME'           Destination activated
      D4C6        URBLTDSD EQU   C'MF'           Destination deactivated
      D4C7        URBLTDSO EQU   C'MG'           Destination opened
      D4C8        URBLTDSC EQU   C'MH'           Destination closed
      D4C9        URBLTDSU EQU   C'MI'           Destination unavailable
      D4D1        URBLTDSE EQU   C'MJ'           Destination error
      D4D2        URBLTDSF EQU   C'MK'           Destination full
      D4D3        URBLTFLA EQU   C'ML'           File activated
      D4D4        URBLTFLD EQU   C'MM'           File deactivated
      D4D5        URBLTIQO EQU   C'MN'           Input queue opened
      D4D6        URBLTIQC EQU   C'MO'           Input queue closed
      D4D7       URBLTRPO EQU   C'MP'          Replication pool overflow in Reptor
      D4D8        URBLTRDL EQU   C'MQ'           Replication data lost
      D4D9        URBLTSUS EQU   C'MR'           TLOGing suspended
      D4E2        URBLTTSF EQU   C'MS'           TLOG Space Allocation Failure
      D4E3        URBLTRFR EQU   C'MT'           Refresh requested
      D4E4        URBLTRFF EQU   C'MU'           Refresh failed
      D4E5        URBLTRFE EQU   C'MV'           Resource refreshed
      D4E6        URBLTRFC EQU   C'MW'           Refresh completed
      D4E7        URBLTRFX EQU   C'MX'           Refresh timed out
      D4E8        URBLTRFA EQU   C'MY'           Refresh aborted by user
```

```
      D4E9        URBLTRFN EQU   C'MZ'           Refresh not done (resource not ↩
changed)
      D4F0       URBLTDSR EQU   C'M0'          Retry open of destination attempted
      D4F1       URBLTIQR EQU   C'M1'          Retry open of input queue attempted
      D5E9        URBLTSCH EQU   C'NZ'           Highest state-change event
                  *
                  *        End of URBL types
                  *
00010             URBLTIME DS    XL8           Date/time (STCK) when event occurred
00018              URBLDNAM DS    CL8            Destination associated with event
00020              URBLSNAM DS    CL8            Subscription associated with event
                  *
00028              URBLPTYP DS    H              Type of TLOG data (at URBLDATA):
                  *
                 *       The URBT, URBR, URBI, and URBS structures are available in
                 *       separate macros. All other structures are mapped using the
                   *       DSECTs below.
                  *
      0000        URBLPNOD EQU   0               No additional data
      0001        URBLPUBT EQU   1              Output transaction              ↩
(URBT)
      0002        URBLPUBR EQU   2              Output record                   ↩
(URBLR)
      0003        URBLPULD EQU   3              Output record image             ↩
(URBLD)
      0004        URBLPULT EQU   4              Input transaction               ↩
(URBLT)
      0005        URBLPULR EQU   5              Input record                    ↩
(URBLQ)
      0006        URBLPULB EQU   6              Input record image              ↩
(URBLB)
      0007        URBLPRPI EQU   7               Reptor status change   ↩
(URBLS->URBLSRPN)
      0008        URBLPFIL EQU   8               File status change     ↩
(URBLS->URBLSDBI)
      0009        URBLPIQN EQU   9               Input Q status change  ↩
(URBLS->URBLSIQN)
      000A        URBLPUBI EQU   10             User request input data         ↩
(URBI)
      000B        URBLPUBS EQU   11             Status/response output data     ↩
(URBS)
      000C        URBLPULI EQU   12             User request input buffer       ↩
(URBLI)
      000D        URBLPCNT EQU   13              Event count            ↩
(URBLS->URBLSCNT)
      000E        URBLPFLT EQU   14             Filter event data               ↩
(URBLF)
      000F        URBLPEUS EQU   15             Extended status/response output  ↩
(URBLE)
      0010        URBLPRFR EQU   16              Refresh resource info  ↩
(URBLS->URBLSRRQ)
      0011        URBLPFCI EQU   17             Filter condition info           ↩
```

```
(URBLC)
      0012        URBLPADA EQU   18              ADABAS error info               ↩
(URBLA)
                  *
0002A                      DS    XL22            Reserved
                  *
00040             URBLDATA DS    0D              Data related to event follows
      0040        URBLL    EQU   *-URBL          Length of TLOG base record
                  ↩
*-----------------------------------------------------------------------*
                  *                                                       ↩
      *
                 *        DSECT to map an ADABAS error record written to the TLOG ↩
     *
                  *                                                       ↩
      *
                  ↩
*-----------------------------------------------------------------------*
00000 00000 00020 URBLA    DSECT ,              Output record image prefix
00000             URBLAEYE DS    CL6             URBLA eye-catcher 'URBLA '
00006             URBLASDB DS    H               Source database ID
00008             URBLASFN DS    H               Source file number
0000A             URBLATDB DS    H               Target database ID
0000C             URBLATFN DS    H               Target file number
0000E             URBLARSP DS    H               Response code
00010             URBLAREA DS    F               Reason code
00014             URBLAISN DS    F               ISN
00018             URBLACMD DS    CL2             Command  code
0001A                      DS    XL6             Reserved
      0020        URBLAL   EQU   *-URBLA         Length of ADABAS error record
                  ↩
*-----------------------------------------------------------------------*
                  *                                                       ↩
      *
                  *        DSECT to map an input record image             ↩
      *
                  *        (before subscription processing)               ↩
      *
                  *                                                       ↩
      *
                  ↩
*-----------------------------------------------------------------------*
00000 00000 00030 URBLB    DSECT ,              Input record image prefix
00000             URBLBEYE DS    CL6             URBLB eye-catcher 'URBLB '
00006             URBLBTYP DS    C               Type of image:
      00C1        URBLBTYA EQU   C'A'              Compressed after image of data ↩
storage
      00C2        URBLBTYB EQU   C'B'              Compressed before image of data ↩
storage
     00D2       URBLBTYK EQU   C'K'            Compressed before image of primary ↩
key
      00C4        URBLBTYD EQU   C'D'              Special type FDT
```

```
      00C6        URBLBTYF EQU   C'F'            Special type FCB
00007                      DS    X              Reserved
00008             URBLBTTM DS    XL8           Transaction timestamp (commit time)
00010            URBLBRSN DS    F             Record sequence number in transaction
00014             URBLBISQ DS    F               Sequence number of image
00018             URBLBPAO DS    F               Offset into payload represented by
                  *                               this record
0001C             URBLBPAL DS    F               Payload length in this record
00020                      DS    XL16            Reserved
00030             URBLBPAY DS    0D             Payload follows - Compressed record
      0030        URBLBL   EQU   *-URBLB        Length of input record image prefix
                  ↩
*-------------------------------------------------------------------*
                  *                                                   ↩
      *
                  *        DSECT to map the filter condition information written ↩
for    *
                 *        a filter match, nomatch, or ignore TLOG event.         ↩
      *
                  *                                                   ↩
      *
                  ↩
*-------------------------------------------------------------------*
00000 00000 00050 URBLC    DSECT ,            Filter condition match/nomatch/ignore
00000             URBLCEYE DS    CL6             URBLC eye-catcher 'URBLC '
00006             URBLCFNM DS    CL8             Filter name
0000E             URBLCGRP DS    H               Filter group number
00010             URBLCNUM DS    H               Filter condition number
00012             URBLCCIN DS    C               Condition indicator:
      00C9        URBLCCIG EQU   C'I'              Condition ignored
      00D4        URBLCCMA EQU   C'M'              Condition matched
      00D5        URBLCCNM EQU   C'N'              Condition not matched
00013             URBLCUPT DS    C               Update type:
      00C4        URBLCUPD EQU   C'D'              Delete
      00C9        URBLCUPI EQU   C'I'              Insert
      00D9        URBLCUPR EQU   C'R'              Initial-state (refresh)
      00E4        URBLCUPU EQU   C'U'              Update
00014             URBLCISN DS    F               ISN
00018             URBLCPAL DS    F               Payload length in this record
                  *
                 *       The following describes the source field of the condition.
                  *
0001C             URBLCSFN DS    CL2             Source field name
0001E             URBLCSPE DS    H               Source PE number
00020             URBLCSMU DS    H               Source MU number
00022             URBLCSLN DS    H               Source field length
                  *                               0 if data not included in payload
00024             URBLCDBI DS    H               Database ID
00026             URBLCNUC DS    H               Nucleus ID in cluster
00028             URBLCFNR DS    H               File number
0002A             URBLCSIM DS    C               Source image:
      00C1        URBLCSAI EQU   C'A'              After  image of data storage
```

```
      00C2        URBLCSBI EQU   C'B'            Before image of data storage
      00D2        URBLCSKY EQU   C'K'            Before image of primary key
0002B             URBLCSTY DS    C              Source type:
      00C1        URBLCSTA EQU   C'A'            Alpha
      00C2        URBLCSTB EQU   C'B'            Binary
      00C6        URBLCSTF EQU   C'F'            Fixed point
      00C7        URBLCSTG EQU   C'G'            Floating point
      00D7        URBLCSTP EQU   C'P'            Packed
      00E4        URBLCSTU EQU   C'U'            Unpacked
      00E6        URBLCSTW EQU   C'W'            Wide character
                  *
0002C             URBLCCND DS    CL2            Condition
0002E             URBLCCLK DS    XL8            Transaction commit timestamp (STCK)
00036             URBLCRUT DS    XL8            Record updated timestamp (STCK)
0003E             URBLCSBG DS    XL2            Begin byte in source field
                  *
                  *        The following fields are filled out only if the target
                  *        of the condition is a field.
                  *
00040             URBLCTFN DS    CL2            Target field name
00042             URBLCTPE DS    H              Target PE number
00044             URBLCTMU DS    H              Target MU number
00046             URBLCTLN DS    H              Target field length
                  *                              0 if data not included in payload
                  *                                or no target field specified
00048             URBLCTIM DS    C              Target image:
      00C1        URBLCTAI EQU   C'A'            After  image of data storage
      00C2        URBLCTBI EQU   C'B'            Before image of data storage
      00D2        URBLCTKY EQU   C'K'            Before image of primary key
00049             URBLCTTY DS    C              Target type:
      00C1        URBLCTTA EQU   C'A'            Alpha
      00C2        URBLCTTB EQU   C'B'            Binary
      00C6        URBLCTTF EQU   C'F'            Fixed point
      00C7        URBLCTTG EQU   C'G'            Floating point
      00D7        URBLCTTP EQU   C'P'            Packed
      00E4        URBLCTTU EQU   C'U'            Unpacked
      00E6        URBLCTTW EQU   C'W'            Wide character
                  *
0004A             URBLCTBG DS    XL2            Begin byte in target field
0004C                      DS    XL4            Reserved
00050             URBLCPAY DS    0D             Payload follows - Source/target ↩
value(s)
      0050        URBLCL   EQU   *-URBLC        Length of filter condition prefix
                  ↩
*-----------------------------------------------------------------------*
                  *                                                       ↩
      *
                  *        DSECT to map an output record image            ↩
      *
                  *        (after subscription processing)                ↩
      *
                  *                                                       ↩
```

```
      *
                  ↩
*-------------------------------------------------------------------*
00000 00000 00030 URBLD    DSECT ,              Output record image prefix
00000              URBLDEYE DS    CL6            URBLD eye-catcher 'URBLD '
00006                       DS    XL2            Reserved
00008             URBLDTTM DS    XL8           Transaction timestamp (commit time)
00010            URBLDRSN DS    F             Record sequence number in transaction
00014             URBLDISQ DS    F              Sequence number of image
00018             URBLDPAO DS    F              Offset into payload represented by
                  *                              this record
0001C             URBLDPAL DS    F              Payload length in this record
00020                       DS    XL16           Reserved
00030             URBLDPAY DS    0D             Payload follows - URBD
      0030        URBLDL   EQU   *-URBLD        Length of output record image prefix
                  ↩
*-------------------------------------------------------------------*
                  *                                                 ↩
      *
                  *        DSECT to map the extended Status/Response event. This  ↩
      *
                  *        is written when the URBS data must be written over     ↩
      *
                  *        multiple TLOG records.                                 ↩
      *
                  *                                                               ↩
      *
                  ↩
*-------------------------------------------------------------------*
00000 00000 00030 URBLE    DSECT ,              Extended Status/response prefix
00000              URBLEEYE DS    CL6            URBLE eye-catcher 'URBLE '
00006                       DS    XL2            Reserved
00008             URBLETIM DS    XL8            Time from field URBSTIME
00010             URBLETOT DS    F              Total length of input buffer ↩
(URBS+data)
00014            URBLEPAO DS    F             Offset into URBS+data represented by
                  *                              this record
00018             URBLEPAL DS    F              Payload length in this record
0001C                       DS    XL20           Reserved
00030             URBLEPAY DS    0D             Payload follows - URBS+data
      0030        URBLEL   EQU   *-URBLE        Length of extended status/response ↩
prefix
                  ↩
*-------------------------------------------------------------------*
                  *                                                 ↩
      *
                  *        DSECT to map the data written for a filter event       ↩
      *
                  *                                                               ↩
      *
                  ↩
*-------------------------------------------------------------------*
```

```
00000 00000 00020 URBLF    DSECT ,               Filter event data
00000             URBLFEYE DS    CL6             URBLF eye-catcher 'URBLF '
00006             URBLFREA DS    C               Reason for exclusion:
      00C5        URBLFREQ EQU   C'E'            SFREPLICATENOTCHANGED specified and
                  *                                AI=BI
      00D4        URBLFRFF EQU   C'M'             Matched an exclude filter
      00D5        URBLFRNI EQU   C'N'             Did not match any include filter
      00E7        URBLFRUX EQU   C'X'             Subscription user exit
00007                      DS    X               Reserved
00008             URBLFFNM DS    CL8             Name of filter (if any)
00010             URBLFGID DS    H               Filter group ID (if any)
00012                      DS    XL14            Reserved
      0020        URBLFL   EQU   *-URBLF         Length of filter event data
                  ↩
*-------------------------------------------------------------------*
                  *                                                  ↩
      *
                  *        DSECT to map the input buffer received from a user,    ↩
      *
                 *       written for request received and request rejected events ↩
    *
                  *        (before and after conversion)                          ↩
      *
                  *                                                               ↩
      *
                  ↩
*-------------------------------------------------------------------*
00000 00000 00020 URBLI    DSECT ,               User input buffer prefix
00000             URBLIEYE DS    CL6             URBLI eye-catcher 'URBLI '
00006                      DS    XL2             Reserved
00008             URBLIRSP DS    F               Zero for 'request received' event
                  *                              Reason code for 'request rejected'
0000C             URBLITOT DS    F               Total length of input buffer
00010             URBLIPAL DS    F               Length of this portion
00014             URBLIPAO DS    F               Offset this portion represents
00018                      DS    XL8             Reserved
00020             URBLIPAY DS    0D              Payload follows - URBH, URBI
      0020        URBLIL   EQU   *-URBLI         Length of user input buffer prefix
                  ↩
*-------------------------------------------------------------------*
                  *                                                  ↩
      *
                  *        DSECT to map an input record                           ↩
      *
                  *                                                               ↩
      *
                  ↩
*-------------------------------------------------------------------*
00000 00000 00040 URBLQ    DSECT ,               Input record prefix
00000             URBLQEYE DS    CL6             URBLQ eye-catcher 'URBLQ '
00006             URBLQIMT DS    C               Type of before image:
      0040        URBLQIMA EQU   C' '             No before image
```

```
      00C2        URBLQIMB EQU   C'B'           Compressed data storage before image
       00C6        URBLQIMF EQU   C'F'             FCB (within pseudo-transaction)
       00D2        URBLQIMK EQU   C'K'             Compressed primary key value
                   *                              If initial-state process:
       00C5        URBLQIME EQU   C'E'              End of file reached
       00D9        URBLQIMR EQU   C'R'              Process stopped due to
                   *                                 response code (in URBLQRSP/SUB)
       00E2        URBLQIMS EQU   C'S'              Process stopped on request ↩
(HALT/STOP)
00007               URBLQUPT DS    C              Type of update:
       00C4        URBLQUPD EQU   C'D'              Record deleted
       00C9        URBLQUPI EQU   C'I'              Record inserted
       00E4        URBLQUPU EQU   C'U'              Record updated
                   *                              Within pseudo-transactions:
       00C6        URBLQUPF EQU   C'F'              FCB and FDT
       00D3        URBLQUPL EQU   C'L'              Report of replication data loss
       00D9        URBLQUPR EQU   C'R'              Record from initial-state process
       00E8        URBLQUPY EQU   C'Y'              File replication reactivated
       00E9        URBLQUPZ EQU   C'Z'              File replication deactivated
00008                       DS    XL2            Reserved
0000A               URBLQFNR DS    H              File number
0000C               URBLQISN DS    F              ISN
00010               URBLQTTM DS    XL8           Transaction timestamp (commit time)
00018               URBLQRSN DS    F             Record sequence number in transaction
0001C               URBLQRSP DS    H               Response code
0001E               URBLQSUB DS    XL2             Response subcode
00020               URBLQFMT DS    XL8             STCK of 1st modification of same ↩
FNR/ISN
00028               URBLQLMT DS    XL8            STCK of last modification of same ↩
FNR/ISN
00030               URBLQBFZ DS    F               Adabas command counter
00034                       DS    XL12            Reserved
       0040        URBLQL   EQU   *-URBLQ         Length of input record prefix
                   ↩
*------------------------------------------------------------------------*
                   *                                                      ↩
       *
                   *        DSECT to map an output record                 ↩
       *
                   *                                                      ↩
       *
                   ↩
*------------------------------------------------------------------------*
00000 00000 00020 URBLR    DSECT ,              Output record prefix
00000               URBLREYE DS    CL6            URBLR eye-catcher 'URBLR '
00006                       DS    XL2            Reserved
00008               URBLRTTM DS    XL8           Transaction timestamp (commit time)
00010                       DS    XL16           Reserved
00020               URBLRPAY DS    0D             Payload follows - URBR
       0020        URBLRL   EQU   *-URBLR         Length of output record prefix
                   ↩
*------------------------------------------------------------------------*
```

```
                 *                                                              ↩
      *
                 *       DSECT to map records written for state change events   ↩
      *
                 *                                                              ↩
      *
                 ↩
*-------------------------------------------------------------------*
00000 00000 00020 URBLS    DSECT ,              State change event data
00000             URBLSEYE DS    CL6            URBLS eye-catcher 'URBLS '
00006                      DS    XL2            Reserved
00008             URBLSESD DS    0C             Event specific data follows here
                 ↩
*-------------------------------------------------------------------*
                 *                                                              ↩
      *
                 *       Data written for a Reptor-related state change event   ↩
      *
                 *                                                              ↩
      *
                 ↩
*-------------------------------------------------------------------*
00008 00008 00008          ORG   URBLSESD
00008             URBLSRPN DS    CL8            Reptor address space name
00010             URBLSRPS DS    C              Start indicator:
      00D9        URBLSRPR EQU   C'R'            Restart after abnormal shutdown
      00E2        URBLSRPI EQU   C'S'            Start after normal shutdown
00011                      DS    XL15           Reserved
      0020        URBLSRPL EQU   *-URBLS
                 ↩
*-------------------------------------------------------------------*
                 *                                                              ↩
      *
                 *       Data written for a file-related state change event     ↩
      *
                 *                                                              ↩
      *
                 ↩
*-------------------------------------------------------------------*
00020 00020 00008          ORG   URBLSESD
00008             URBLSDBI DS    H              Database ID
0000A             URBLSFNR DS    H              File number
0000C                      DS    XL20           Reserved
      0020        URBLSDFL EQU   *-URBLS
                 ↩
*-------------------------------------------------------------------*
                 *                                                              ↩
      *
                 *        Data written for an input queue-related state change ↩
event   *
                 *                                                              ↩
      *
```

```
                  ↩
*-------------------------------------------------------------------*
00020 00020 00008          ORG   URBLSESD
00008              URBLSIQN DS    CL8            Input queue name
00010                       DS    XL16           Reserved
      0020         URBLSIQL EQU   *-URBLS
                  ↩
*-------------------------------------------------------------------*
                  *                                                  ↩
      *
                 *       Data written for state change events requiring counters ↩
     *
                  *                                                  ↩
      *
                  ↩
*-------------------------------------------------------------------*
00020 00020 00008          ORG   URBLSESD
00008              URBLSCNT DS    F              Count of events
0000C                       DS    XL20           Reserved
      0020         URBLSCNL EQU   *-URBLS
                  ↩
*-------------------------------------------------------------------*
                  *                                                  ↩
      *
                 *        Data written for a refresh of a specific resource      ↩
      *
                  *                                                  ↩
      *
                  ↩
*-------------------------------------------------------------------*
00020 00020 00008          ORG   URBLSESD
00008              URBLSRRQ DS    C              Refresh request:
      00C1         URBLSRRA EQU   C'A'            Add a resource
      00C4         URBLSRRD EQU   C'D'            Delete a resource
      00D4         URBLSRRM EQU   C'M'            Modify a resource
                  *
00009              URBLSRTY DS    C              Type of resource:
      00C1         URBLSRTA EQU   C'A'            Initial state
      00C4         URBLSRTD EQU   C'D'            Destination
      00C6         URBLSRTF EQU   C'F'            Full refresh
      00C7         URBLSRTG EQU   C'G'            Globals
      00C9         URBLSRTI EQU   C'I'            Input queue
      00D9         URBLSRTR EQU   C'R'            Resend buffer
      00E2         URBLSRTS EQU   C'S'            Subscription
                  *
0000A              URBLSRNM DS    CL8            Resource name
00012                       DS    XL14           Reserved
      0020         URBLSRFL EQU   *-URBLS
                  ↩
*-------------------------------------------------------------------*
                  *                                                  ↩
      *
```

```
                  *        End of state change record                          ↩
      *
                  *                                                            ↩
      *
                  ↩
*-------------------------------------------------------------------*
00020 00020 00020          ORG   ,
     0020       URBLSL   EQU   *-URBLS       Maximum length of state change event ↩
data
                  ↩
*-------------------------------------------------------------------*
                  *                                                            ↩
      *
                  *        DSECT to map an input transaction                   ↩
      *
                  *                                                            ↩
      *
                  ↩
*-------------------------------------------------------------------*
00000 00000 00060 URBLT    DSECT ,              Input transaction data
00000              URBLTEYE DS    CL6            URBLT eye-catcher 'URBLT '
00006                       DS    XL2            Reserved
00008              URBLTDBI DS    H              Originating Adabas database ID
0000A              URBLTNUI DS    H              Originating Adabas nucleus ID
0000C              URBLTGUI DS    XL28           Originating 28-byte user ID
00028              URBLTTBT DS    XL8            STCK at transaction begin
                   *                              (time of first replicated update)
00030             URBLTTTM DS    XL8           STCK at transaction end (commit time)
00038              URBLTTRI DS    F              Transaction ID (by nucleus)
0003C              URBLTTUT DS    F              Transaction ID (by user)
00040              URBLTREC DS    F              Number of records in transaction
00044             URBLTINS DS    F             Initial-state request relative number
00048              URBLTRSI DS    C              Indicator for possible double ↩
delivery:
      00E8         URBLTRSY EQU   C'Y'             Data for the same transaction
                   *                               may have been sent previously
00049              URBLTPTI DS    C              Pseudo-transaction indicator:
      00C3         URBLTSC5 EQU   C'C'             C5 user data pseudo-transaction
      00C4        URBLTSDA EQU   C'D'            File deactivation pseudo-transaction
      00C6         URBLTSEU EQU   C'F'             FCB/FDT pseudo-transaction
      00C9         URBLTSIS EQU   C'I'             Initial-state pseudo-transaction
      00D3         URBLTSLO EQU   C'L'             Lost replication data ↩
pseudo-transaction
                   *
0004A              URBLTSRT DS    C              Indicates whether or not the ↩
transaction
                   *                             records have been sorted by file, ↩
ISN,
                   *                               and relative number:
      00E8         URBLTSRY EQU   C'Y'             Transaction has been sorted
      00D5         URBLTSRN EQU   C'N'             Transaction has not been sorted
0004B                       DS    X              Reserved
```

```
0004C              URBLTSID DS    CL8            Security system user ID
00054                       DS    XL12           Reserved
      0060         URBLTL   EQU   *-URBLT        Length of input transaction data
      F0F1         URBLVERC EQU   URBLVER1              Current version - Initial ↩
release
```

URBP

```
                          URBP     ,              Replication buffer - Parm Subs. ↩
Exit
00000 00000 00014 URBP     DSECT
                  ↩
*-----------------------------------------------------------------------*
                  *        URBP -- Subscription user exit parameter list          ↩
      *
                  *                                                               ↩
      *
                  *        DSECT URBP is used by the Event Replicator for Adabas. ↩
      *
                  *                                                               ↩
      *
                  *        DSECT URBP describes the parameter address list passed ↩
      *
                  *        to the Reptor subscription user exit.                  ↩
      *
                  ↩
*-----------------------------------------------------------------------*
                  *                                                               ↩
      *
                   *        During Reptor session start, the user exit is called ↩
once    *
                  *        with the Initialize function.                          ↩
      *
                  *                                                               ↩
      *
                 *        While the Reptor is running, the exit is called with the ↩
    *
                  *        Process-record function for every record that is being ↩
      *
                 *        replicated. The exit may filter the record (suppress its ↩
    *
                  *       shipment to the destination(s)) or modify its contents. ↩
     *
                  *                                                               ↩
      *
                   *        During Reptor shutdown (normal termination only), the ↩
exit   *
                  *        is called once with the Terminate function.            ↩
      *
```

```
                  *                                                               ↩
      *
                  ↩
*-----------------------------------------------------------------------*
                  *                                                               ↩
      *
                  *        The subscription user exit is called using a           ↩
      *
                  *        standard BALR 14,15 interface, as follows:             ↩
      *
                  *                                                               ↩
      *
                  *        On entry,                                              ↩
      *
                  *           R1  -> URBP parameter address list                  ↩
      *
                  *           R13 -> Standard 72-byte register save area          ↩
      *
                  *           R14 -> Return point                                 ↩
      *
                  *           R15 -> User exit entry point                        ↩
      *
                  *                                                               ↩
      *
                  *        On exit,                                               ↩
      *
                  *           information may be returned in the first parameter  ↩
      *
                  *                                                               ↩
      *
                  ↩
*-----------------------------------------------------------------------*
00000             URBPURBX DS    A              Addr of URBX parameter block
                  *
                  *  Only for the Process-record function (URBXFUNC=URBXFREC):
                  *
00004             URBPURBT DS    A              Addr of URBT
00008             URBPURBR DS    A              Addr of URBR
0000C             URBPABI  DS    A              Addr of URBD of before image
                  *                              Zero if no before image
00010             URBPAAI  DS    A              Addr of URBD of after image
                  *                              Zero if no after image
                  *
                *  The user exit may change the before/after image record contents
               *  at URBD.URBDDATA and may decrease their lengths at URBD.URBDLEND.
                  *  However, it must not increase the record data lengths. If the
                *  exit sets a length to zero, the corresponding record image will
                  *  not be shipped to the destination(s).
                  *
                *  The last supplied address in the URBP parameter address list is
                  *  marked with the high bit set.
```

```
                  *
      0014        URBPL    EQU   *-URBP         DSECT length
```

URBQ

```
000000 00000 00008  URBQ     DSECT                                          ↩
                     ↩
*-------------------------------------------------------------------*
                    *        URBQ -- User Exit Program Parameter Block      ↩
        *
                    *                                                       ↩
        *
                    *        DSECT URBQ is used by the Adabas Replication Extract ↩
        *
                    *        utility (ADARPE).                              ↩
        *
                    *                                                       ↩
        *
                   *        DSECT URBQ describes the parameter address list passed ↩
      *
                    *        from the extract utility to a user exit program.  ↩
        *
                    *                                                       ↩
        *
                    *        The user exit is called using a standard BALR 14,15  ↩
        *
                    *        interface, as follows:                         ↩
        *
                    *                                                       ↩
        *
                    *        On entry,                                      ↩
        *
                    *           R1  -> URBQ parameter address list          ↩
        *
                    *           R13 -> Standard 72-byte register save area  ↩
        *
                    *           R14 -> Return point                         ↩
        *
                    *           R15 -> User exit entry point                ↩
        *
                    *                                                       ↩
        *
                    *        On exit,                                       ↩
        *
                   *          information may be returned in the first parameter ↩
       *
                    *                                                       ↩
        *
```

```
                    ↩
*-------------------------------------------------------------------*
000000              URBQURBZ DS    A              Address of URBZ parameter block ↩

000004              URBQABUF DS    A               Address of output buffer for ↩
record
                    *                                                  ↩

                   *  The last supplied address in the URBQ parameter address list ↩
is
                    *  marked with the high bit set.                   ↩

                    *                                                  ↩

       00008        URBQL    EQU   *-URBQ         DSECT length                    ↩
```

URBR

```
                           URBR     ,                Replication buffer - Record
00000 00000 00040 URBR     DSECT
                   ↩
*-------------------------------------------------------------------*
                   *        URBR -- Record element                    ↩
      *
                   *                                                  ↩
      *
                   *        DSECT URBR is used by the Adabas Replication Facility. ↩
      *
                   *                                                  ↩
      *
                   *        DSECT URBR contains information related to one record. ↩
      *
                   *                                                  ↩
      *
                   ↩
*-------------------------------------------------------------------*
00000              URBREYE  DS    CL4            URBR eye-catcher 'URBR'
00004              URBRLEN  DS    F              Length of URBR
                   *
00008              URBRRSNR DS    F              Record sequence number within ↩
transaction
0000C              URBRDCNT DS    H              Count of data elements (URBDs)
                   *                              for this record
                   *
0000E              URBRFNR  DS    H              Original Adabas file number
00010              URBRISN  DS    F              Original Adabas ISN
00014              URBRTIME DS    XL8            Time (STCK) of last modification
                   *
```

```
0001C             URBRTYP  DS    C              Type of update:
      00C4        URBRTYPD EQU   C'D'            Delete
      00C9        URBRTYPI EQU   C'I'            Insert
      00D9        URBRTYPR EQU   C'R'            Initial State (refresh)
      00E4        URBRTYPU EQU   C'U'            Update
                  *
0001D             URBRRSND DS    C              Indicator for possible double ↩
delivery:
      00E8        URBRRSNY EQU   C'Y'            Data for same record may have been
                  *                               sent previously
                  *
0001E             URBRRSP  DS    H              Response code
00020             URBRSUBC DS    XL4            Subcode information
00024             URBRERRC DS    CL8            Other error code information
                  *                             No record data (URBD) is shipped ↩
if the
                 *                            record incurred a nonzero response ↩
code
                 *                            (URBRRSP) or non-blank other error ↩
code
                  *                              (URBRERRC) during replication
                  *                              processing.
                  *
0002C             URBRDCU  DS    C              Indicator for possible delete ↩
conversion:
      00E8        URBRDCUY EQU   C'Y'            Delete converted to update
                  *
0002D             URBRUC   DS    C              Indicator for possible update ↩
conversion:
      00C4        URBRUCD  EQU   C'D'            Update converted to delete
      00C9        URBRUCI  EQU   C'I'            Update converted to insert
                  *
0002E                      DS    XL18           Reserved area
00040                      DS    0D
      0040        URBRL    EQU   *-URBR         DSECT length
```

URBS

```
                           URBS     ,                Replication buffer - ↩
Status/response
00000 00000 00140 URBS     DSECT                                          ↩

                  ↩
*-------------------------------------------------------------------*
                  *        URBS -- Reptor status/response element          ↩
      *
                  *                                                        ↩
      *
                  *        DSECT URBS is used by the Adabas Replication Facility. ↩
```

```
      *
                  *                                                          ↩
      *
                  *        DSECT URBS contains status/response information sent   ↩
      *
                  *        from the Reptor to a Target Application.          ↩
      *
                  *                                                          ↩
      *
                  ↩
*-------------------------------------------------------------------*
00000             URBSEYE  DS    CL4           URBS eye-catcher 'URBS'          ↩

00004             URBSLEN  DS    F             Length of URBS + related data + ↩
filler
00008             URBSRTOK DS    XL8           Request ID token (from original ↩
request),
                 *                           zero if unsolicited Reptor message ↩

                  *                                                          ↩

00010            URBSRT   DS    CL4          Request type (blank if no request): ↩

      C1E3        URBSRTST EQU   C'STAT'        Subscription status information  ↩

     E2E3       URBSRTIS EQU   C'INST'       Initial-state processing information ↩

      C1D5        URBSRTTA EQU   C'TRAN'        Prior-transaction processing ↩
information
      D5C4        URBSRTOD EQU   C'OPND'        Open a destination               ↩

      E2C4        URBSRTCD EQU   C'CLSD'        Close a destination              ↩

                  *                                                          ↩

00014            URBSST   DS    CL4          Reptor status/response information: ↩

     C9E3       URBSSTIN EQU   C'INIT'       Initial-state processing commenced ↩

      D9D6        URBSSTER EQU   C'ERRO'        Initial-state erroneous request ↩
rejected
     D7D3       URBSSTCM EQU   C'CMPL'       Initial-state processing completed ↩

      C2E2        URBSSTSU EQU   C'SUBS'        Subscription status information  ↩

      D9E3        URBSSTUP EQU   C'STRT'        Normal Reptor session start      ↩

      E2E3        URBSSTAR EQU   C'REST'        Reptor restart after abnormal end ↩

     D9D4       URBSSTDN EQU   C'TERM'       Normal Reptor session termination ↩

      C1C3        URBSSTDA EQU   C'DEAC'        Replication deactivated          ↩
```

```
      C1C3        URBSSTRA EQU   C'REAC'         Replication reactivated          ↩
      E2D7        URBSSTTR EQU   C'TRSP'          Response to prior-transaction ↩
request
      E2E3        URBSSTLO EQU   C'LOST'         Possibly lost replication data   ↩
      C4C1        URBSSTC5 EQU   C'C5DA'          Replication user data from C5 ↩
command
      C4E2        URBSSTLS EQU   C'LODS'         ADALOD LOAD started              ↩
      C4C5        URBSSTLE EQU   C'LODE'         ADALOD LOAD ended                ↩
      D3E2        URBSSTRS EQU   C'RPLS'         Replay process (ADARPL) started  ↩
      D3C5        URBSSTRE EQU   C'RPLE'         Replay process (ADARPL) ended    ↩
      C6D9        URBSSTRF EQU   C'REFR'         Resource refreshed               ↩
      E5E2        URBSSTSS EQU   C'SAVS'         ADASAV RESTORE started           ↩
      E5C5        URBSSTSE EQU   C'SAVE'         ADASAV RESTORE ended             ↩
      E2E2       URBSSTGS EQU   C'RESS'        ADARES REGENERATE/BACKOUT started ↩
      E2C5        URBSSTGE EQU   C'RESE'         ADARES REGENERATE/BACKOUT ended  ↩
      C4E2        URBSSTUS EQU   C'UPDS'         ADALOD UPDATE started            ↩
      C4C5        URBSSTUE EQU   C'UPDE'         ADALOD UPDATE ended              ↩
      E3C9       URBSSTAU EQU   C'AUTI'        Adabas utility service replication ↩
      D6E2        URBSSTCL EQU   C'CLOS'         Destination closed               ↩
      C5D5        URBSSTOP EQU   C'OPEN'         Destination opened               ↩
      C5C3        URBSSTAS EQU   C'ASEC'         Adabas security replication      ↩
      D6D5        URBSSTSN EQU   C'SLON'         SLOG turned on for destination   ↩
      D6C6        URBSSTSF EQU   C'SLOF'         SLOG turned off for destination  ↩
      C4D6        URBSSTDO EQU   C'SLDO'         SLOG for DB-related input turned ↩
on
      C4C6        URBSSTDF EQU   C'SLDF'         SLOG for DB-related input turned ↩
off
     C4E2       URBSSTDS EQU   C'SLDS'        SLOG for DB-related input suspended ↩
      C4D9        URBSSTDR EQU   C'SLDR'         SLOG for DB-related input resumed ↩
      C4C4        URBSSTDD EQU   C'SLDD'         SLOG for DB-related input disabled ↩
```

```
      E4D3        URBSSTFL EQU   C'DFUL'         Destination full                ↩
      D9D9        URBSSTDE EQU   C'DERR'         Destination error               ↩
      C3D6        URBSSTCO EQU   C'DBCO'         Database connected              ↩
      C4C9        URBSSTDI EQU   C'DBDI'         Database disconnected           ↩
      D5D7        URBSSTNP EQU   C'DBNP'         Database not present            ↩
     D4D7        URBSSTDC EQU   C'DCMP'         Decompression error: AI, BI, FAI, ↩
FBI,
                  *                               or KEY                         ↩
      C6C7        URBSSTRG EQU   C'REFG'         Refresh Globals                 ↩
      D6D7        URBSSTQO EQU   C'IQOP'         Input Queue opened              ↩
      C3D3        URBSSTQC EQU   C'IQCL'         Input Queue closed              ↩
                  *                                                              ↩
00018             URBSTIME DS    XL8            Time (STCK) when this URBS was ↩
created
                  *                                                              ↩
00020             URBSRSP  DS    F              Response code for request       ↩
00024             URBSSUBC DS    F              Subcode for request             ↩
00028             URBSERRI DS    CL8            Other pertinent error information ↩
                  *                                                              ↩
00030             URBSINAM DS    CL8           Initial-state name (CMPL/ERRO/INIT ↩
                  *                              message)                        ↩
00038             URBSSNAM DS    CL8            Subscription name ↩
(ASEC/C5DA/DCMP/DEAC/
                 *                             LOST/REAC/REFR/RPLE/RPLS/SUBS/TRSP ↩
                  *                              message)                        ↩
00040             URBSDNAM DS    CL8            Destination name ↩
(DEAC/DERR/FULL/REAC/
                  *                              REFR/SLOF/SLON message)         ↩
                  *                                                              ↩
00048             URBSPTIM DS    XL8            Time (STCK) when last transaction ↩
```

```
was
                  *                                processed for the subscription   ↩
                  *                                (SUBS message)                   ↩
00050             URBSTTIM DS    XL8               Time (STCK) of transaction commit ↩
                  *                                (ASEC/AUTI/C5DA/SUBS message)    ↩
00058             URBSTSNR DS    F                 Transaction sequence number within ↩
                  *                               subscription/destination (SUBS/TRSP ↩
msg)
                  *                                                                 ↩
0005C             URBSDBFN DS    OF                DBID and FNR ↩
(ASEC/CMPL/DCMP/DEAC/ERRO/
                  *                               INIT/LODS/LODE/LOST/REAC/SAVS/SAVE/ ↩
                  *                                UPDS/UPDE message)               ↩
0005C             URBSDBID DS     H                 Database ID ↩
(DBCO/DBDI/DBNP/RPLS/RPLE
                  *                                 messages)                       ↩
0005E             URBSFNR  DS     H                 File number                     ↩
                  *                                                                 ↩
00060             URBSLENH DS    F                 Length of URBS                   ↩
00064            URBSLEND DS    F                 Length of related data (at URBSDATA) ↩
                  *                                                                 ↩
00068             URBSUTOK DS    H                 Utility token ↩
(CMPL/ERRO/INIT/LODS/LODE/
                  *                                 RPLS/RPLE/SAVS/SAVE/UPDS/UPDE ↩
message)
                  *                                                                 ↩
0006A            URBSORIG DS    C                Origin of status (ASEC/C5DA message): ↩
      00C1        URBSORIA EQU   C'A'              Coming from Adabas nucleus       ↩
      00D9        URBSORIR EQU   C'R'              Coming from ADARPL REPLAY        ↩
                  *                                                                 ↩
0006B                     DS    XL5               Reserved area                    ↩
00070             URBSIQNM DS    CL8               Input Queue name (IQCL/IQOP/REFR ↩
```

```
message)
00078                     DS    XL8           Reserved area                    ↩
                  *                                                            ↩
00080             URBSDATA DS     0D           Payload data (ASEC/C5DA/RPLE/RPLS ↩
                  *                             message)                       ↩
                  *                             Note: For the RPLS/RPLE message, ↩
                 *                            URBSDATA contains a list of 2-byte ↩
file
                  *                            numbers of the file(s) associated ↩
with
                   *                             the replay process for the ↩
subscription
                  *                             given in URBSSNAM              ↩
                  *                                                            ↩
      0080        URBSL    EQU   *-URBS         DSECT length                   ↩
```

URBT

```
                          URBT     ,               Replication buffer - Transaction
00000 00000 00080 URBT     DSECT
                  ↩
*-------------------------------------------------------------------*
                  *        URBT -- Transaction element                      ↩
      *
                  *                                                         ↩
      *
                  *       DSECT URBT is used by the Adabas Replication Facility. ↩
      *
                  *                                                         ↩
      *
                   *        DSECT URBT contains information related to one ↩
transaction.  *
                  *                                                         ↩
      *
                  ↩
*-------------------------------------------------------------------*
00000             URBTEYE  DS    CL4            URBT eye-catcher 'URBT'
00004             URBTLEN  DS    F              Length of URBT
                  *
00008             URBTSNAM DS    CL8            Subscription name
00010             URBTTSNR DS    F              Transaction sequence number within
```

```
                  *                                  subscription/destination
00014             URBTRCNT DS    F                  Count of records (URBRs) in ↩
transaction
00018             URBTTTIM DS    XL8                Time (STCK) at transaction commit
00020             URBTPTIM DS    XL8                Time (STCK) at end of subscription
                  *                                  processing
                  *
00028             URBTDBID DS    H                  Originating Adabas database ID
0002A             URBTNUCI DS    H                  Originating Adabas nucleus ID
0002C             URBTGUID DS    XL28               Originating 28-byte user ID
                  *
00048             URBTRPID DS    H                  Reptor target ID (DBID parameter)
0004A             URBTRPNI DS    H                  Reptor nucleus ID (not yet used) ↩
=> zero
                  *
0004C            URBTUSRV DS    CL2                User subscription version indicator,
                  *                                   set from parameter SVERSION in the
                  *                                   subscription definition
0004E             URBTRSND DS    C                  Indicator for possible double ↩
delivery:
      00E8        URBTRSNY EQU   C'Y'                Data for same transaction may have
                  *                                   been sent previously
0004F             URBTINST DS    C                  Indicator for initial state:
      00E8        URBTINSY EQU   C'Y'                Data sent by initial-state process
                  *
00050             URBTRTOK DS    XL8                Request ID token (if response to ↩
request
                 *                                 from target application => URBIRTOK)
                  *
00058             URBTCONT DS    C                  Indicator for transaction ↩
continuation:
      00E8        URBTCONY EQU   C'Y'                More data to follow in the next ↩
message
                  *                                   for the same transaction
                  *
                  *  The fields URBTARC, URBTACOD and URBTWCOD contain the ↩
subscription
                  *  parameters SARC, SACODE and SWCODE, respectively.
                  *
00059             URBTARC  DS    X                  Architecture of transaction data
                  *
0005A             URBTPTRN DS    C                  Indicator for prior-transaction ↩
request:
     00E8        URBTPTRY EQU   C'Y'                Resending data from prior transaction
                  *
0005B             URBTSORT DS    C                  Indicates whether or not the ↩
transaction
                 *                                  records have been sorted by file, ↩
ISN,
                  *                                   and relative number:
      00E8        URBTSORY EQU   C'Y'                Transaction has been sorted
      00D5        URBTSORN EQU   C'N'                Transaction has not been sorted
```

```
                 *
0005C            URBTACOD DS    F              Encoding of alpha fields
00060            URBTWCOD DS    F              Encoding of wide fields
                 *
00064           URBTUTOK DS    H             Utility token, if transaction comes ↩
from
                *                              ADARPL REPLAY or ADALOD LOAD/UPDATE
                 *
00066            URBTORIG DS    C              Origin of transaction:
      00C1       URBTORIA EQU   C'A'             Coming from Adabas nucleus
      00C7       URBTORGG EQU   C'G'             Coming from ADARES
      00D3       URBTORIL EQU   C'L'             Coming from ADALOD LOAD
      00D9       URBTORIR EQU   C'R'             Coming from ADARPL REPLAY
      00E2       URBTORIS EQU   C'S'             Coming from ADASAV
      00E4       URBTORIU EQU   C'U'             Coming from ADALOD UPDATE
                 *
00067                     DS    X              Reserved area
                 *
00068            URBTSUID DS    CL8            Security system user ID
                 *
00070                     DS    XL16           Reserved area
00080                     DS    0D
      0080       URBTL    EQU   *-URBT         DSECT length
```

URBU

```
000000 00000 00050  URBU     DSECT                                          ↩

                     ↩
*-------------------------------------------------------------------*
                    *        URBU -- ADARPE Extract Header User Element     ↩
        *
                    *                                                       ↩
        *
                   *       DSECT URBU is used by the Adabas Replication Facility. ↩
      *
                    *                                                       ↩
        *
                   *       DSECT URBU contains data describing the extract created ↩
     *
                   *        from ADARPE.  It is written as the first record of the ↩
      *
                    *        output file of ADARPE when HEADER=YES is specified.  ↩
        *
                    *                                                       ↩
        *
                    ↩
*-------------------------------------------------------------------*
000000              URBUEYE  DS    CL4            URBU eye-catcher 'URBU'     ↩
```

```
000004              URBULEN  DS    F              Length of URBU                 ↩
                    *                                                           ↩
000008              URBUVERS DS    CL2            URBU version indicator:        ↩
       0F0F1        URBUVER1 EQU   C'01'           First version                 ↩
                    *                             Future releases may allow a new ↩
layout
                    *                              of the ADARPE output file. In ↩
this case
                   *                            URBUVERS will be set to a different ↩
                    *                             value.                        ↩
                    *                                                           ↩
00000A              URBUNAME DS    CL8             Extract name (ADARPE NAME ↩
parameter)
000012              URBUTIME DS    XL8            Time of execution (STCK)       ↩
00001A             URBUDIST DS    CL22          Local time (YYYY-MM-DD HH:MM:SS.hh) ↩
                    *                                                           ↩
000030              DS    XL32           Reserved area
000050              DS    0D
       00050        URBUL    EQU   *-URBU         DSECT length                  ↩
```

URBX

```
                          URBX     ,              Replication buffer - Par1 Subs. ↩
Exit
00000 00000 00060 URBX     DSECT
                   ↩
*-------------------------------------------------------------------*
                   *        URBX -- Subscription user exit parameter block     ↩
      *
                   *                                                          ↩
      *
                   *        DSECT URBX is used by the Adabas Replication Facility. ↩
      *
                   *                                                          ↩
      *
                   *        DSECT URBX describes the first parameter passed to the ↩
```

```
      *
                  *        Reptor subscription user exit.                        ↩
      *
                  ↩
*-------------------------------------------------------------------*
00000             URBXEYE  DS    CL4      (in)  URBX eye-catcher 'URBX'
00004             URBXLEN  DS    F        (in)  Length of URBX
                  *
00008            URBXVERS DS    CL2     (in)  Version indicator for exit parameters:
      F0F1        URBXVER1 EQU   C'01'           First version
                 *                            Future releases may allow a new layout
                  *                              of the subscription user exit ↩
parameters
                  *                             (URBP and URBX DSECTs). In this case
                  *                              URBXVERS will be set to a different
                  *                              value.
0000A             URBXVERH DS    CL2      (in)  Version indicator for URB* DSECTs,
                  *                               corresponding to URBHVERS
                  *
0000C             URBXUSER DS    A    (in/out)  Word for use by user exit:
                 *                             Initially (Initialize function), zero
                 *                             Then, value set by exit in previous ↩
call
                  *
00010             URBXFUNC DS    X        (in)  Function code:
      0001        URBXFINI EQU   X'01'           Initialize
      0002        URBXFTRM EQU   X'02'           Terminate
      0003        URBXFREC EQU   X'03'           Process record
                  *
00011             URBXRETC DS    X       (out)  Return code:
      0000        URBXRSHP EQU   X'00'           Ship record (default setting)
      0001        URBXRFLT EQU   X'01'           Filter record (other nonzero
                  *                               values cause filtering, too)
                  *
00012             URBXJNAM DS    CL8      (in)  Reptor job name
                  *
0001A             URBXERRC DS    CL8     (out)  Other error code information:
                  *                              Passed to destination in ↩
URBR.URBRERRC
                  *                              (default setting is blanks)
                  *
00022             URBXRETD DS    X       (out)  Return indicator for delete ↩
processing:
      0000       URBXRDEL EQU   X'00'          Process as delete (default setting)
      0001        URBXRUPD EQU   X'01'           Convert delete to update
                  *
                  *
00023             URBXRETU DS    X       (out)  Return indicator for update ↩
processing:
      0001        URBXRUTI EQU   X'01'           Convert update to insert
      0002        URBXRUTD EQU   X'02'           Convert update to delete
                  *
```

```
00024                    DS    XL58          Reserved area
00060                    DS    0D
      0060        URBXL    EQU   *-URBX        DSECT length
                  *
```

URBZ

```
000000 00000 00040  URBZ     DSECT                                          ↩

                     ↩
*--------------------------------------------------------------------*
                     *        URBZ -- User Exit Program Parameter Block     ↩
        *
                     *                                                      ↩
        *
                     *        DSECT URBZ is used by the Adabas Replication Extract ↩
        *
                     *        utility (ADARPE).                             ↩
        *
                     *                                                      ↩
        *
                    *        DSECT URBZ describes the first parameter passed to the ↩
      *
                     *        extract utility user exit.                    ↩
        *
                     *                                                      ↩
        *
                     ↩
*--------------------------------------------------------------------*
000000               URBZEYE  DS    CL4      (in)  URBZ eye-catcher 'URBZ'  ↩

000004               URBZLEN  DS    F        (in)  Length of URBZ           ↩

                     *                                                      ↩

000008               URBZVERS DS    CL2      (in)  Version indicator for exit ↩
parameters:
       0F0F1         URBZVER1 EQU   C'01'           First version           ↩

                     *                              Future releases may allow a new ↩
layout
                     *                               of the extract user exit ↩
parameters.
                    *                              In this case URBZVERS will be set ↩
to a
                     *                               different value.       ↩

00000A               URBZVERH DS    CL2     (in)  Version indicator for URB* DSECTs, ↩
```

```
                   *                              corresponding to URBHVERS      ↩
                   *                                                             ↩
00000C             URBZUSER DS    A    (in/out)  Word for use by user exit:     ↩
                   *                             Initially (Initialize function), ↩
zero
                   *                               Then, value set by exit in ↩
previous call
                   *                                                             ↩
000010             URBZDDN  DS    CL8      (in)  Output DD/DLBL name            ↩
000018             URBZDDML DS    F        (in)  Maximum record length of output ↩
file
00001C              URBZDDWL DS    F    (in/out)  Length to write, modifiable by ↩
user exit
                   *                                                             ↩
000020             URBZDTYP DS    X        (in)  Type of data provided in buffer: ↩
       00001       URBZTYPL EQU   1                URBL                          ↩
       00002       URBZTYPT EQU   2                URBT                          ↩
       00003       URBZTYPR EQU   3                URBLR/URBR                    ↩
       00004       URBZTYPD EQU   4                URBLD/URBD                    ↩
       00005       URBZTYPS EQU   5                URBS                          ↩
       00006       URBZTYPE EQU   6                URBLE/URBS                    ↩
       00007       URBZTYPP EQU   7                Raw payload                   ↩
                   *                                                             ↩
000021             URBZFUNC DS    X        (in)  User exit function code:       ↩
       00001       URBZFINI EQU   X'01'            Initialize user exit          ↩
       00002       URBZFTRM EQU   X'02'            Terminate user exit           ↩
       00003       URBZFOUT EQU   X'03'            Process output data           ↩
                   *                                                             ↩
000022             URBZRETC DS    X       (out)  User exit return code:         ↩
                   *                             Initialize/Terminate functions: ↩
```

```
       00000        URBZITOK EQU   X'00'            Success                        ↩

       00001        URBZITER EQU   X'01'            Error - will result in ADARPE ↩
Error-127
                    *                              Process function:              ↩

       00000       URBZRWRT EQU   X'00'            Write record (default setting) ↩

       00001        URBZRSKP EQU   X'01'            Skip record                    ↩

       00002        URBZRERR EQU   X'02'            Error - will result in ADARPE ↩
Error-127
                    *                                                             ↩

000023              URBZERRC DS    CL8     (out)  Other error code information -  ↩

                    *                               will be printed in Error-127 ↩
message as
                   *                               "User Exit error code=xxxxxxxx" ↩

                    *                                                             ↩

00002B                       DS    XL21           Reserved area                  ↩

       00040        URBZL    EQU   *-URBZ         DSECT length                   ↩
```

5 Event Replicator Client Requests

- Specifying the URBH 56
- Destination Open and Close Requests 57
- Initial-State Requests 60
- Prior-Transaction Requests 76
- Status Requests 80

Clients can send requests to the Event Replicator Server by sending messages to an Event Replicator input queue.

An input message is composed of a combination of:

- a message header defined by the URBH DSECT
- one or more input elements defined by the URBI DSECT.

The following requests are defined:

- Destination open and close
- Initial-state processing
- Prior-transaction processing.
- Status inquiry on subscription

Specifying the URBH

The input message starts with a message header defined by the URBH DSECT. Set the elements in the URBH DSECT as described in the following table. Set all other elements in the URBH DSECT to the default value of binary zeroes or blanks according to the element type.

Element	Setting
URBHEYE	Set to 'URBH'. This value may be specified in either EBCDIC or ASCII. If this field contains 'URBH' in ASCII, all character fields in the URBH and URBI(s) will be translated from ASCII to EBCDIC.
URBHLEN	Set to the length of the URBH DSECT. This length is given in variable URBHL.
URBHVERS	Set to '01'.
URBHBORD	Set to the binary value of one. On big-endian machines, this will have the value X'0001'. On little endian-machines, this will have the value X'0100'. If this field is contains X'0100', all binary fields in the URBH and URBI(s) will be byte-swapped before being used by the Event Replicator Server.
URBHLENT	Set to the total message length. This includes the length of the URBH plus the length of each subsequent URBI.
URBHTIME	Optionally set this variable to the store clock time when the message was sent. The value in this variable within an input message is not referenced by the Event Replicator Server.
URBHNAME	Optionally set this variable to the name of the sender. The value in this variable within an input message is not referenced by the Event Replicator Server.

Destination Open and Close Requests

A destination open or close request will open or close a destination from your application program. Multiple requests can be made using multiple URBIs following a URBH header. The status message (URBS) contains the result of processing.

- URBI Fields
- URBS Fields
- Destination Open Request Example

URBI Fields

For a destination open or close request, set the elements in the URBI DSECT as described in the following table. Values may be provided for only these fields. All unused binary fields must be set to zero and all unused character fields must be set to blanks.

Element	Setting
URBIEYE	Set to 'URBI'.
URBILEN	Set to URBIL.
URBILENH	Set to URBIL.
URBIRT	Set to 'OPND' to request that the destination be opened; set to 'CLSD' to request that the destination be closed. One of these settings is required.
URBIDNAM	Supply the definition name of the destination that should be opened or closed. This element is required.
URBIRNAM	Optionally supply a response destination name.
URBIRTOK	Optionally supply the eight-byte token that allows your client program to recognize different open or close destination requests.

URBS Fields

Status messages for a destination open or close request contain the following information:

Status Element	Description
URBSRT	"OPND" appears if a destination was opened; "CLSD" appears if a destination was closed.

Destination Open Request Example

This example requests a destination named BROUT2 be closed. Destination BROUT1 is used as the response destination. The following topics are included in this section.

- Create and send the destination close request
- Review the output in the BROUT1 destination

Create and send the destination close request

The following application code requests that destination BROUT2 be closed:

```
rr=newMessage(bb.send_buffer)  # start a new message beginning with URBH

rr.requestCloseDestination(  # followed by URBI
   token=TOKEN001,
   rnam=BROUT1,
   dnam=BROUT2)

bb.send_length=rr.endMessage() # end message with URBHLENT = total message size
                               # and set length for messaging system

bb.send()                      # send message
```

This is the resulting message with URBH and URBI settings generated from the close destination request that is sent to the Event Replicator input queue:

```
0000 E4D9C2C8 00000040 F0F10001 000000A0  *URBH... 01.....-*
0010 00000010 BBBBBBBB BBBBBBBB 00000000  *................*
0020 C3D3D6E2 C4C5E2E3 00000000 00000000  *CLOSDEST........*
0030 00000000 00000000 00000000 00000000  *................*
0040 E4D9C2C9 00000060 00000060 00000000  *URBI...-...-....*
0050 E3D6D2C5 D5F0F0F1 C2D9D6E4 E3F14040  *TOKEN001BROUT1  *
0060 C3D3E2C4 00000000 40404040 40404040  *CLSD....        *
0070 40404040 40404040 C2D9D6E4 E3F24040  *        BROUT2  *
0080 00000000 00000000 00000000 00000000  *................*
0090 00000000 00000000 00000000 00000000  *................*
```

Review the output in the BROUT1 destination

The following depicts the received message at destination BROUT1 for normal completion of the URBI:

```
0000 E4D9C2C8 00000040 F0F10001 000000C0  *URBH... 01......*
0010 00000001 BB680AF2 C1C27F60 115C0000  *.......2AB"-.*..*
0020 D9C5D7E3 D6D94040 00000000 00000000  *REPTOR.........*
0030 00000000 00000000 00000000 00000000  *................*
0040 E4D9C2E2 00000080 E3D6D2C5 D5F0F0F1  *URBS....TOKEN001*
0050 C3D3E2C4 C3D3D6E2 BB680AF2 C1C18060    *CLSDCLOSC .2AA.-*
0060 00000000 00000000 40404040 40404040  *........        *
0070 40404040 40404040 40404040 40404040  *                *
0080 C2D9D6E4 E3F24040 00000000 00000000  *BROUT2  ........*
0090 00000000 00000000 00000000 00000000  *................*
00A0 00000080 00000000 00004000 00000000  *.......... .....*
00B0 00000000 00000000 00000000 00000000  *................*
```

which can be interpreted as:

```
URBH element at offset X'0000'
     urbheye  = "URBH"
     urbhlen  = 64
     urbhvers = "01"
     urbhbord = X'0001'
     urbhlent = 192
     urbhmsnr = 1
     urbhtime = 2004-06-22 12:22:34.789927
     urbhrpid = 4444
     urbhrpni = 0
     urbhname = "REPTOR"

URBS element at offset X'0040'
     urbseye  = "URBS"
     urbslen  = 128
     urbsrtok = "TOKEN001"
     urbsrt   = "CLSD"
     urbsst   = "CLOS" Destination closed
     urbstime = 2004-06-22 12:22:34.789912
     urbsrsp  = 0
     urbssubc = 0
     urbserri = "        "
     urbsinam = "        "
     urbssnam = "        "
     urbsdnam = "BROUT2  "
     urbsptim = 0
     urbsttim = 0
     urbstsnr = 0
     urbsdbid = 0
     urbsfnr  = 0
```

Initial-State Requests

An initial-state request initiates the initial-state processing in the Event Replicator and the nucleus. During initial-state processing, the nucleus reads the selected records and passes them to the Event Replicator Server. The Event Replicator decompresses the records depending on the subscription format and sends the data to the assigned output destinations.

Records can be selected in one of the following manners:

- The complete file can be selected.
- Records are selected from the file based on an ISN list.
- Records are selected from the file based on specified selection criteria.

Initial-state requests must be supported by initial-state definitions. Initial-state definitions are specified in the Adabas Event Replicator Subsystem or by `INITIALSTATE` parameters in the Event Replicator Server startup job. For more information about initial-state definitions, read about maintaining initial-state definitions using the Adabas Event Replicator Subsystem (read *Adabas Event Replicator Subsystem User's Guide*). For information about the `INITIALSTATE` parameter, read *Event Replicator Initialization Parameters* in *Event Replicator for Adabas Reference Guide*

- URBI Fields
- ISN List Format
- Responses
- Initial-State Request Examples

URBI Fields

For an initial-state request, set the elements in the URBI DSECT as described in the following table. If the initial-state definition includes more than one file-database ID definition for the request (a list of them), the initial-state request must contain a URBI for each file.

Element	Setting
URBIEYE	Set to 'URBI'.
URBILEN	Set to `URBIL + URBILEND + length of filler for alignment`
URBILENH	Set to URBIL.
URBILEND	Set to `length(selection criteria or ISN list)`. This element should be set to zero ('0') in an initial-state request where the all records in the file are selected by the initial-state definition.
URBIRT	Set to 'INST'
URBINAM	Supply the initial-state definition name as defined in the Adabas Event Replicator Subsystem or by the `INITIALSTATE` parameter in the Event Replicator Server startup job. The value of this element must be the same for all URBIs in a message.

Element	Setting
URBIFNR	Supply the file number of the Adabas input file.
URBIDBID	Supply the database ID of the Adabas database containing the input file.
URBIRNAM	Supply the name of the response destination definition.
URBIRTOK	Optionally supply the eight-byte token that allows your client program to recognize different initial-state requests. If you specify this element, its value must be the same in all URBIs in a message.

The following three fields should be set if the selection data (URBIDATA) has a different encoding or architecture than the database. The usage is similar to the ACODE, WCODE, and ARC parameters in the record buffer for the Adabas OP command (as described in the Adabas command reference documentation found in *Adabas Command Reference*). These fields may be used only if the Event Replicator Server is UES-enabled. They must be zero for status requests and for initial-state requests:

Element	Description
URBIACOD	Alpha field encoding in selection data.
URBIARC	Architecture flags for selection data.
URBIWCOD	Wide field encoding in selection data.

ISN List Format

If the initial-state definition indicates that an ISN list should be used to select records for processing, the initial-state request must include an ISNLIST with the requested ISNs.

The ISNLIST comes behind the URBI structure. It contains a sequence of individual ISNs (fullword) or an ISN range with three fullwords consisting of:

- A range indicator X'FFFFFFFF'
- The starting ISN
- The ending ISN

An ISN list must conform to the following rules:

1. The same ISN must not be specified more than once with one exception: in a range the starting and ending ISN can be the same.
2. An individual ISN may not lie within an ISN range
3. ISN ranges must not overlap
4. starting ISN <= ending ISN

Violation of these rules will return to the response destination a URBS status message with response code 131, subcode 16.

Should a requested ISN not be present, the system reacts as follows:

- For an individual ISN, the initial-state payload will contain a URBR with response code 113, subcode 0 without record data, even if the ISN is beyond TOPISN or outside of MINISN through MAXISN.
- Only existent ISNs will be returned for the ISN range.

Responses

If the Event Replicator Server discovers an error in the initial-state request, it sends a URBS status message INST/ERRO to the response destination.

Otherwise, the URBS status message INST/INIT is sent to the response and related output destinations and the initial-state request will be forwarded to the nucleus.

Should any error occur with the nucleus (for example if the file is locked), the Event Replicator Server will terminate the initial-state processing and send a final URBS status message INST/CMPL with the error information to the response and output destinations. In contrast, when no error occurs, the nucleus will send initial-state data wrapped in pseudo transactions.

At the end, the Event Replicator Server will notify the response and related output destinations with a URBS status message INST/CMPL.

Initial-State Request Examples

This section provides several examples of initial-state requests.

- Simple Request
- Request Using Selection Criteria and Conversion

Simple Request

This example requests data based on the initial-state definition named ICOLOR.

- Specify the replication parameters
- Create and send the initial-state request
- Review the output in the OUT1 destination

- Review invalid initial-state request

Specify the replication parameters

The initial-state definition named ICOLOR indicates that file 4 of database 10006 should be used for initial-state processing. The subscription definition COLOR defines the output format and destination that will receive the initial-state data.

```
ADARPD INITIALSTATE NAME=ICOLOR,IFILE=4,IDBID=10006
*
ADARPD SUBSCRIPTION ↩
NAME=COLOR,SFILE=4,SFDBID=10006,SFBAI='AA,AB.',SDESTINATION='OUT1'
```

Create and send the initial-state request

The following application code submits the initial-state request.

```
rr=newMessage(bb.send_buffer)  # start a new message beginning with URBH

rr.requestInitialState(  # followed by URBI
   token=TOKENTOK,
   rnam=OUT1,
   inam=ICOLOR,
   dbid=10006,
   fnr=4)

bb.send_length=rr.endMessage() # end message with URBHLENT = total message size
                               # and set length for messaging system

bb.send()                      # send message
```

This is the resulting message with URBH and URBI settings generated from the initial-state request that is sent to the Event Replicator input queue:

```
    0000 E4D9C2C8 00000040 F0F10001 000000A0 URBH... 01......
    0010 00000000 00000000 00000000 00000000 ................
    0020 C1849489 95404040 00000000 00000000 Admin   ........
    0030 00000000 00000000 00000000 00000000 ................
    0040 E4D9C2C9 00000060 00000060 00000000 URBI...-...-....
    0050 E3D6D2C5 D5E3D6D2 D6E4E3F1 40404040 TOKENTOKOUT1
    0060 C9D5E2E3 27160004 C9C3D6D3 D6D94040 INST....ICOLOR
    0070 40404040 40404040 40404040 40404040
    0080 00000000 00000000 00000000 00000000 ................
         1 identical line(s) suppressed
```

Review the output in the OUT1 destination

The following messages are received on the OUT1 destination specified by the replication parameters:

1. The Event Replicator Server accepts the request and sends the INST/INIT status message. URBH is followed by URBS.

```
    0000 E4D9C2C8 00000040 F0F10001 000000C0 URBH... 01......
    0010 00000067 BB57F5D2 9C176D82 27170000 ......5K.._b....
    0020 D9C5D7E3 D6D94040 00000000 00000000 REPTOR  ........
    0030 00000000 00000000 00000000 00000000 ................
    0040 E4D9C2E2 00000080 E3D6D2C5 D5F14040 URBS....TOKEN1
    0050 C9D5E2E3 C9D5C9E3 BB57F5D2 8A8E6E62 INSTINIT..5K..>.
    0060 00000000 00000000 40404040 40404040 ........
    0070 C9C3D6D3 D6D94040 40404040 40404040 ICOLOR
    0080 40404040 40404040 00000000 00000000         ........
    0090 00000000 00000000 00000000 27160004 ................
    00A0 00000000 00000000 00000000 00000000 ................
         1 identical line(s) suppressed
```

   which can be interpreted as:

```
URBH element at offset X'0000'
     urbheye  = "URBH"
     urbhlen  = 64
     urbhvers = "01"
     urbhbord = X'0001'
     urbhlent = 192
     urbhmsnr = 103
     urbhtime = 2004-06-09 17:22:52.308854
     urbhrpid = 10007
     urbhrpni = 0
     urbhname = "REPTOR"

URBS element at offset X'0040'
     urbseye  = "URBS"
     urbslen  = 128
     urbsrtok = "TOKEN1"
     urbsrt   = "INST"
     urbsst   = "INIT" Initial-state processing started
     urbstime = 2004-06-09 17:22:52.237030
     urbsrsp  = 0
     urbssubc = 0
     urbserri = "        "
     urbsinam = "ICOLOR"
     urbssnam = "        "
     urbsdnam = "        "
     urbsptim =
     urbsttim =
     urbstsnr = 0
```

```
urbsdbid = 10006
urbsfnr  = 4
```

2. The initial-state data comes from the database; file 4 has only three records. The nucleus wraps the data in a pseudo-transaction (URBT) containing records (URBR) and after images (URBD).

```
0000 E4D9C2C8 00000040 F0F10001 00000250 URBH... 01.....&
0010 00000069 BB57F5D2 CF429E80 27170000 ......5K........
0020 D9C5D7E3 D6D94040 00000000 00000000 REPTOR  ........
0030 00000000 00000000 00000000 00000000 ................
0040 E4D9C2E3 00000070 C3D6D3D6 D9404040 URBT....COLOR
0050 00000000 00000003 BB57F5D2 AEE40D22 ..........5K.U..
0060 BB57F5D2 B68E2102 27160000 11111111 ..5K............
0070 22222222 BB57F5D2 93F61021 00000000 ......5K¦6......
0080 27170000 00040000 27170000 404040E8 ............   Y
0090 E3D6D2C5 D5F14040 40000000 00000000 TOKEN1   ........
00A0 00000000 00000000 00000000 00000000 ................
00B0 E4D9C2D9 00000040 00000001 00010004 URBR... ........
00C0 00000002 BB57F5D2 AEA661A2 D9400000 ......5K.w/sR ..
00D0 00000000 40404040 40404040 00000000 ....        ....
00E0 00000000 00000000 00000000 00000000 ................
00F0 E4D9C2C4 00000040 00000020 00000020 URBD... ........
0100 00000001 C1000000 00000000 00000000 ....A...........
0110 C7D9C5C5 D5404040 40404040 40404040 GREEN
0120 F2F0F0F4 F0F5F2F5 F1F8F3F5 F4F6F1F0 2004052518354610
0130 E4D9C2D9 00000040 00000002 00010004 URBR... ........
0140 00000003 BB57F5D2 AEE3BE02 D9400000 ......5K.T..R ..
0150 00000000 40404040 40404040 00000000 ....        ....
0160 00000000 00000000 00000000 00000000 ................
0170 E4D9C2C4 00000040 00000020 00000020 URBD... ........
0180 00000001 C1000000 00000000 00000000 ....A...........
0190 C2D3E4C5 40404040 40404040 40404040 BLUE
01A0 F2F0F0F4 F0F5F2F5 F1F8F3F5 F4F6F1F1 2004052518354611
01B0 E4D9C2D9 00000040 00000003 00010004 URBR... ........
01C0 00000004 BB57F5D2 AEE3FF62 D9400000 ......5K.T..R ..
01D0 00000000 40404040 40404040 00000000 ....        ....
01E0 00000000 00000000 00000000 00000000 ................
01F0 E4D9C2C4 00000040 00000020 00000020 URBD... ........
0200 00000001 C1000000 00000000 00000000 ....A...........
0210 E8C5D3D3 D6E64040 40404040 40404040 YELLOW
0220 F2F0F0F4 F0F5F2F5 F1F8F3F5 F4F6F1F2 2004052518354612
0230 E4D9C2C5 00000020 C3D6D3D6 D9404040 URBE....COLOR
0240 00000000 00000000 00000000 00000000 ................
```

which can be interpreted as:

```
URBH element at offset X'0000'
     urbheye  = "URBH"
     urbhlen  = 64
     urbhvers = "01"
     urbhbord = X'0001'
     urbhlent = 592
     urbhmsnr = 105
     urbhtime = 2004-06-09 17:22:52.518441
     urbhrpid = 10007
     urbhrpni = 0
     urbhname = "REPTOR"

URBT element at offset X'0040'
     urbteye  = "URBT"
     urbtlen  = 112
     urbtsnam = "COLOR"
     urbttsnr = 0
     urbtrcnt = 3
     urbtttim = 2004-06-09 17:22:52.385856
     urbtptim = 2004-06-09 17:22:52.417250
     urbtdbid = 10006
     urbtnuci = 0
     urbtguid = X'1111111122222222BB57F5D293F6102100000000027170000000040000'
     urbtrpid = 10007
     urbtrpni = 0
     urbtusrv = "  "
     urbtrsnd = " "
     urbtinst = "Y"
     urbtrtok = "TOKEN1"
     urbtcont = " "


URBR element at offset X'00B0'
     urbreye  = "URBR"
     urbrlen  = 64
     urbrrsnr = 1
     urbrdcnt = 1
     urbrfnr  = 4
     urbrisn  = 2
     urbrtime = 2004-06-09 17:22:52.384870
     urbrtyp  = "R" initial state
     urbrrsnd = " "
     urbrrsp  = 0
     urbrsubc = 0
     urbrerrc = "        "

URBD element at offset X'00F0'
     urbdeye  = "URBD"
     urbdlen  = 64
     urbdlenh = 32
     urbdlend = 32
     urbddsnr = 1
```

```
     urbdtyp  = "A" after image
     ubddata  = "GREEN","2004052518354610"


URBR element at offset X'0130'
     urbreye  = "URBR"
     urbrlen  = 64
     urbrrsnr = 2
     urbrdcnt = 1
     urbrfnr  = 4
     urbrisn  = 3
     urbrtime = 2004-06-09 17:22:52.385851
     urbrtyp  = "R" initial state
     urbrrsnd = " "
     urbrrsp  = 0
     urbrsubc = 0
     urbrerrc = "        "

URBD element at offset X'0170'
     urbdeye  = "URBD"
     urbdlen  = 64
     urbdlenh = 32
     urbdlend = 32
     urbddsnr = 1
     urbdtyp  = "A" after image
     urbddata = "BLUE","2004052518354611"

URBR element at offset X'01B0'
     urbreye  = "URBR"
     urbrlen  = 64
     urbrrsnr = 3
     urbrdcnt = 1
     urbrfnr  = 4
     urbrisn  = 4
     urbrtime = 2004-06-09 17:22:52.385855
     urbrtyp  = "R" initial state
     urbrrsnd = " "
     urbrrsp  = 0
     urbrsubc = 0
     urbrerrc = "        "

URBD element at offset X'01F0'
     urbdeye  = "URBD"
     urbdlen  = 64
     urbdlenh = 32
     urbdlend = 32
     urbddsnr = 1
     urbdtyp  = "A" after image
     urbddata = "YELLOW","2004052518354612"


URBE element at offset X'0230'
```

```
     urbeeye  = "URBE"
     urbelen  = 64
     urbesnam = "COLOR"
     urbetsnr = 0
```

3. At the end of initial-state processing, the Event Replicator Server sends a completion status message URBS/INST/CMPL

```
    0000 E4D9C2C8 00000040 F0F10001 000000C0 URBH... 01......
    0010 0000006A BB57F5D2 CF658501 27170000 ......5K..e.....
    0020 D9C5D7E3 D6D94040 00000000 00000000 REPTOR  ........
    0030 00000000 00000000 00000000 00000000 ................
    0040 E4D9C2E2 00000080 E3D6D2C5 D5F14040 URBS....TOKEN1
    0050 C9D5E2E3 C3D4D7D3 BB57F5D2 B690FF62 INSTCMPL..5K....
    0060 00000000 00000000 40404040 40404040 ........
    0070 C9C3D6D3 D6D94040 40404040 40404040 ICOLOR
    0080 40404040 40404040 00000000 00000000         ........
    0090 00000000 00000000 00000000 27160004 ................
    00A0 00000000 00000000 00000000 00000000 ................
         1 identical line(s) suppressed
```

which can be interpreted as:

```
URBH element at offset X'0000'
     urbheye  = "URBH"
     urbhlen  = 64
     urbhvers = "01"
     urbhbord = X'0001'
     urbhlent = 192
     urbhmsnr = 106
     urbhtime = 2004-06-09 17:22:52.519000
     urbhrpid = 10007
     urbhrpni = 0
     urbhname = "REPTOR"

URBS element at offset X'0040'
     urbseye  = "URBS"
     urbslen  = 128
     urbsrtok = "TOKEN1"
     urbsrt   = "INST"
     urbsst   = "CMPL" Initial-state processing completed
     urbstime = 2004-06-09 17:22:52.417295
     urbsrsp  = 0
     urbssubc = 0
     urbserri = "        "
     urbsinam = "ICOLOR"
     urbssnam = "        "
     urbsdnam = "        "
     urbsptim =
     urbsttim =
     urbstsnr = 0
```

```
     urbsdbid = 10006
     urbsfnr  = 4
```

Review invalid initial-state request

An invalid initial-state request is answered with a status response INST/ERRO to the response destination. The destination file number is not specified correctly.

```
    0000 E4D9C2C8 00000040 F0F10001 000000C0 URBH... 01......
    0010 0000006C BB57FAA1 09D9A980 27170000 ...%.....Rz.....
    0020 D9C5D7E3 D6D94040 00000000 00000000 REPTOR  ........
    0030 00000000 00000000 00000000 00000000 ................
    0040 E4D9C2E2 00000080 E3D6D2C5 D5F14040 URBS....TOKEN1
    0050 C9D5E2E3 C5D9D9D6 BB57FAA1 09D8A720 INSTERRO.....Qx.
    0060 00000083 0000000C 40404040 40404040 ...c....
    0070 C9C3D6D3 D6D94040 40404040 40404040 ICOLOR
    0080 40404040 40404040 00000000 00000000         ........
    0090 00000000 00000000 00000000 27160002 ................
    00A0 00000000 00000000 00000000 00000000 ................
         1 identical line(s) suppressed
```

which can be interpreted as:

```
URBH element at offset X'0000'
     urbheye  = "URBH"
     urbhlen  = 64
     urbhvers = "01"
     urbhbord = X'0001'
     urbhlent = 192
     urbhmsnr = 108
     urbhtime = 2004-06-09 17:44:22.506906
     urbhrpid = 10007
     urbhrpni = 0
     urbhname = "REPTOR"

URBS element at offset X'0040'
     urbseye  = "URBS"
     urbslen  = 128
     urbsrtok = "TOKEN1"
     urbsrt   = "INST"
     urbsst   = "ERRO" Initial-state processing ERROR
     urbstime = 2004-06-09 17:44:22.506890
     urbsrsp  = 131
    urbssubc = 12 INITIALSTATE DBID/FNR definition not found as requested in URBI
     urbserri = "        "
     urbsinam = "ICOLOR"
     urbssnam = "        "
     urbsdnam = "        "
     urbsptim =
     urbsttim =
```

```
    urbstsnr = 0
    urbsdbid = 10006
    urbsfnr  = 2
```

Request Using Selection Criteria and Conversion

This example shows an initial-state request using selection criteria. In addition, since the client is on an Intel x86 machine, the data must be converted for the different machine architecture.

- Specify the replication parameters
- Create and send the initial-state request
- Review the output in the OUT1 destination

Specify the replication parameters

Let's assume the EMPLOYEES file (Adabas example) is replicated to a client application that stores the records on a separate database – for example to provide in-house web access to selected fields for name-telephone look-up.

The subscription definition, EMPLOYEE, defines the following:

- The data is to be delivered to destination OUT1.
- The data is to be converted to ISO8859-1/Latin-1 = ECS code page 819 for alpha fields. The data to be converted to UTF-8 = ECS code page 4091 for wide fields. Integer numbers are to be byte-swapped and floating point data is to be converted to IEEE.
- The file to be replicated is number 9 on database 10006.
- The global format, EMPLTEL, defines the format to be used for the before and after images.

The initial-state definition, IEMPLAA, defines a selection criteria on the PERSONNEL-ID. This is the unique primary key AA. The selection criteria "AA,S,AA." requires the starting and ending values for the primary key. In the example below the selection value "20011000" through "20011100" selects all records in this range – that is 2 records.

```
ADARPD SUBSCRIPTION NAME=EMPLOYEE
ADARPD  SDESTINATION='OUT1'
ADARPD  SACODE=819,SWCODE=4091,SARC=9
ADARPD  SFILE=9,SFDBID=10006
ADARPD  SGFORMATAI=EMPLTEL
*
ADARPD GFORMAT NAME=EMPLTEL  Fields extracted for Telephone List WEBAPP
ADARPD GFB='AA,AC,AD,AE,'    PERSONNEL-ID FIRST-NAME MIDDLE LAST-NAME
ADARPD GFB='AH,8,U,AN,AM,AO,AP.' BIRTH AREA-CODE PHONE DEPT JOB-TITLE
*
ADARPD INITIALSTATE NAME=IEMPLAA,IFILE=9,IDBID=10006,SELCRIT='AA,S,AA.'
*
```

Create and send the initial-state request

The following application code submits the initial-state request.

```
rr=newMessage(bb.send_buffer)  # start a new message beginning with URBH

rr.requestInitialState(  # followed by URBI
   token=EMPLTOKN,
   rnam=OUT1,
   inam=IEMPLTEL,dbid=10006,fnr=4,
   arc=9,acode=819,wcode=4091)

bb.send_length=rr.endMessage() # end message with URBHLENT = total message size
     # and set length for messaging system

bb.send()       # send message
```

This is the resulting message with URBH and URBI settings generated from the initial-state request that is sent to the Event Replicator input queue:

```
    0000 55524248 40000000 30310100 B0000000 URBH@...01......
    0010 00000000 00000000 00000000 00000000 ................
    0020 41646D69 6E202020 00000000 00000000 Admin   ........
    0030 00000000 00000000 00000000 00000000 ................
    0040 55524249 70000000 60000000 10000000 URBIp...........
    0050 454D504C 544F4B4E 4F555432 20202020 EMPLTOKNOUT2
    0060 494E5354 16270900 49454D50 4C414120 INST.'..IEMPLAA
    0070 20202020 20202020 20202020 20202020
    0080 33030000 FB0F0000 09000000 00000000 3...............
    0090 00000000 00000000 00000000 00000000 ................
    00A0 32303031 31303030 32303031 31313030 2001100020011100
```

Review the output in the OUT1 destination

The following messages are received on the OUT1 destination specified by the replication parameters:

1. The Event Replicator Server accepts the request and sends the INST/INIT status message. URBH is followed by URBS.

```
    0000 55524248 40000000 30310100 C0000000 URBH@...01......
    0010 08000000 42CE8C88 B2BA7DBB 17270000 ....B........'..
    0020 52455054 4F522020 00000000 00000000 REPTOR  ........
    0030 00000000 00000000 00000000 00000000 ................
    0040 55524253 80000000 454D504C 544F4B4E URBS....EMPLTOKN
    0050 494E5354 494E4954 626B8888 B2BA7DBB INSTINITbk......
    0060 00000000 00000000 20202020 20202020 ........
    0070 49454D50 4C414120 20202020 20202020 IEMPLAA
    0080 20202020 20202020 00000000 00000000         ........
    0090 00000000 00000000 00000000 16270900 .............'..
```

```
    00A0 00000000 00000000 00000000 00000000 ................
         1 identical line(s) suppressed
```

which can be interpreted as:

```
URBH element at offset X'0000'
     urbheye  = "URBH"
     urbhlen  = 64
     urbhvers = "01"
     urbhbord = X'0001'
     urbhlent = 192
     urbhmsnr = 8
     urbhtime = 18:20:41.098444
     urbhrpid = 10007
     urbhrpni = 0
     urbhname = "REPTOR"

URBS element at offset X'0040'
     urbseye  = "URBS"
     urbslen  = 128
     urbsrtok = "EMPLTOKN"
     urbsrt   = "INST"
     urbsst   = "INIT" Initial-state processing started
     urbstime = 18:20:41.098374
     urbsrsp  = 0
     urbssubc = 0
     urbserri = "        "
     urbsinam = "IEMPLAA"
     urbssnam = "        "
     urbsdnam = "        "
     urbsptim =
     urbsttim =
     urbstsnr = 0
     urbsdbid = 10006
     urbsfnr  = 9
```

2. Here is the initial-state data message:

```
    0000 55524248 40000000 30310100 90020000 URBH@...01......
    0010 09000000 803223B8 B2BA7DBB 17270000 .....2#.......'..
    0020 52455054 4F522020 00000000 00000000 REPTOR  ........
    0030 00000000 00000000 00000000 00000000 ................
    0040 55524254 70000000 454D504C 4F594545 URBTp...EMPLOYEE
    0050 00000000 02000000 62297389 B2BA7DBB ........b)s.....
    0060 00A205A5 B2BA7DBB 16270000 11111111 ........'........
    0070 33333333 BB7DBAB2 889A9A02 00000000 3333............
    0080 27170000 00090000 17270000 20202059 '........'..   Y
    0090 454D504C 544F4B4E 20090000 33030000 EMPLTOKN ...3...
    00A0 FB0F0000 00000000 00000000 00000000 ................
    00B0 55524252 40000000 01000000 01000900 URBR@...........
    00C0 5A020000 428F7289 B2BA7DBB 52200000 Z...B.r.....R ..
```

```
    00D0 00000000 20202020 20202020 00000000 ....        ....
    00E0 00000000 00000000 00000000 00000000 ................
    00F0 55524244 A0000000 20000000 80000000 URBD.... .......
    0100 01000000 41000000 00000000 00000000 ....A...........
    0110 32303031 31303030 424F4220 20202020 20011000BOB
    0120 20202020 20202020 20202020 48454E52             HENR
    0130 59202020 20202020 20202020 20202020 Y
    0140 41444B49 4E534F4E 20202020 20202020 ADKINSON
    0150 20202020 30303334 30353035 30313031     003405050101
    0160 20202833 31362933 38392D39 33323520   (316)389-9325
    0170 2053414C 45323053 414C4553 20504552  SALE2OSALES PER
    0180 534F4E20 20202020 20202020 20202020 SON
    0190 55524252 40000000 02000000 01000900 URBR@...........
    01A0 5B020000 62F57289 B2BA7DBB 52200000 ....b.r.....R ..
    01B0 00000000 20202020 20202020 00000000 ....        ....
    01C0 00000000 00000000 00000000 00000000 ................
    01D0 55524244 A0000000 20000000 80000000 URBD.... .......
    01E0 01000000 41000000 00000000 00000000 ....A...........
    01F0 32303031 31313030 53484952 4C592020 20011100SHIRLY
    0200 20202020 20202020 20202020 502E2020             P.
    0210 20202020 20202020 20202020 20202020
    0220 4D454C4B 414E4F46 46202020 20202020 MELKANOFF
    0230 20202020 30303432 30373230 30313031     004207200101
    0240 20202838 30342933 37362D30 38353020   (804)376-0850
    0250 204D474D 5433304D 414E4147 45522020  MGMT30MANAGER
    0260 20202020 20202020 20202020 20202020
    0270 55524245 20000000 454D504C 4F594545 URBE ...EMPLOYEE
    0280 00000000 00000000 00000000 00000000 ................
```

which can be interpreted as:

```
URBH element at offset X'0000'
     urbheye  = "URBH"
     urbhlen  = 64
     urbhvers = "01"
     urbhbord = X'0001'
     urbhlent = 656
     urbhmsnr = 9
     urbhtime = 18:20:41.293363
     urbhrpid = 10007
     urbhrpni = 0
     urbhname = "REPTOR"

URBT element at offset X'0040'
     urbteye  = "URBT"
     urbtlen  = 112
     urbtsnam = "EMPLOYEE"
     urbttsnr = 0
     urbtrcnt = 2
     urbtttim = 18:20:41.102130
     urbtptim = 18:20:41.215066
     urbtdbid = 10006
```

```
     urbtnuci = 0
     urbtguid = X'1111111133333333BB7DBAB2889A9A020000000002717000000090000'
     urbtrpid = 10007
     urbtrpni = 0
     urbtusrv = "  "
     urbtrsnd = " "
     urbtinst = "Y"
     urbtrtok = "EMPLTOKN"
     urbtcont = " "
     urbtarc  = X'09'
     urbtacod = 819
     urbtwcod = 4091

URBR element at offset X'00B0'
     urbreye  = "URBR"
     urbrlen  = 64
     urbrrsnr = 1
     urbrdcnt = 1
     urbrfnr  = 9
     urbrisn  = 602
     urbrtime = 18:20:41.102120
     urbrtyp  = "R" initial state
     urbrrsnd = " "
     urbrrsp  = 0
     urbrsubc = 0
     urbrerrc = "        "


URBD element at offset X'00F0'
     urbdeye  = "URBD"
     urbdlen  = 160
     urbdlenh = 32
     urbdlend = 128
     urbddsnr = 1
     urbdtyp  = "A" after image
     urbddata =

     0000 32303031 31303030 424F4220 20202020 20011000B0B
     0010 20202020 20202020 20202020 48454E52             HENR
     0020 59202020 20202020 20202020 20202020 Y
     0030 41444B49 4E534F4E 20202020 20202020 ADKINSON
     0040 20202020 30303334 30353035 30313031     003405050101
     0050 20202833 31362933 38392D39 33323520   (316)389-9325
     0060 2053414C 45323053 414C4553 20504552  SALE20SALES PER
     0070 534F4E20 20202020 20202020 20202020 SON

URBR element at offset X'0190'
     urbreye  = "URBR"
     urbrlen  = 64
     urbrrsnr = 2
     urbrdcnt = 1
     urbrfnr  = 9
```

```
     urbrisn  = 603
     urbrtime = 18:20:41.102127
     urbrtyp  = "R" initial state
     urbrrsnd = " "
     urbrrsp  = 0
     urbrsubc = 0
     urbrerrc = "        "

URBD element at offset X'01D0'
     urbdeye  = "URBD"
     urbdlen  = 160
     urbdlenh = 32
     urbdlend = 128
     urbddsnr = 1
     urbdtyp  = "A" after image
     urbddata =

     0000 32303031 31313030 53484952 4C592020 20011100SHIRLY
     0010 20202020 20202020 20202020 502E2020             P.
     0020 20202020 20202020 20202020 20202020
     0030 4D454C4B 414E4F46 46202020 20202020 MELKANOFF
     0040 20202020 30303432 30373230 30313031     004207200101
     0050 20202838 30342933 37362D30 38353020   (804)376-0850
     0060 204D474D 5433304D 414E4147 45522020  MGMT30MANAGER
     0070 20202020 20202020 20202020 20202020


URBE element at offset X'0270'
     urbeeye  = "URBE"
     urbelen  = 32
     urbesnam = "EMPLOYEE"
     urbetsnr = 0
```

3. Here is the initial-state completion message:

```
    0000 55524248 40000000 30310100 C0000000 URBH@...01......
    0010 0A000000 603141B8 B2BA7DBB 17270000 .....1A......'..
    0020 52455054 4F522020 00000000 00000000 REPTOR  ........
    0030 00000000 00000000 00000000 00000000 ................
    0040 55524253 80000000 454D504C 544F4B4E URBS....EMPLTOKN
    0050 494E5354 434D504C 801E08A5 B2BA7DBB INSTCMPL........
    0060 00000000 00000000 20202020 20202020 ........
    0070 49454D50 4C414120 20202020 20202020 IEMPLAA
    0080 20202020 20202020 00000000 00000000         ........
    0090 00000000 00000000 00000000 16270900 ............'..
    00A0 00000000 00000000 00000000 00000000 ................
         1 identical line(s) suppressed
```

which can be interpreted as:

```
URBH element at offset X'0000'
     urbheye  = "URBH"
     urbhlen  = 64
     urbhvers = "01"
     urbhbord = X'0001'
     urbhlent = 192
     urbhmsnr = 10
     urbhtime = 18:20:41.293843
     urbhrpid = 10007
     urbhrpni = 0
     urbhname = "REPTOR"

URBS element at offset X'0040'
     urbseye  = "URBS"
     urbslen  = 128
     urbsrtok = "EMPLTOKN"
     urbsrt   = "INST"
     urbsst   = "CMPL" Initial-state processing completed
     urbstime = 18:20:41.215105
     urbsrsp  = 0
     urbssubc = 0
     urbserri = "        "
     urbsinam = "IEMPLAA"
     urbssnam = "        "
     urbsdnam = "        "
     urbsptim =
     urbsttim =
     urbstsnr = 0
     urbsdbid = 10006
     urbsfnr  = 9
```

Prior-Transaction Requests

If the target application detects a gap in the transaction sequence numbers, a prior-transaction request may be issued to have the Event Replicator Server attempt to send a missing transaction again. Defining a resend buffer and associating it with a subscription enables this capability. Gaps in the transaction number sequence should not occur when a guaranteed-delivery message transport system (such as WebSphere MQ) is being used. A prior-transaction request consists of one **URBH** followed by one **URBI**.

The resend buffer operates in a circular manner. Newer transactions will overwrite older ones. The number of transactions available for retransmission depends on the size of the buffer, the number of subscriptions sharing the buffer, and the size of the transactions. The resend buffer retains the **URBT** and **URBRs** from the transactions, as well as before-image **URBDs**. After-image **URBDs** are not retained.

The subscription name (URBISNAM), destination name (URBIDNAM), and transaction number (URBITSNR) are always required in the prior-transaction request. The transaction will be sent to the single destination identified in the request. Destinations receiving retransmitted transactions must be associated with the subscription and must be for WebSphere MQ or webMethods EntireX messaging systems.

If the optional reply destination (URBIRNAM) is provided, a status message (**URBS**) will be sent to that destination for successful prior-transaction requests.

A **URBS** status message with response code 131 and an appropriate subcode may also be generated for errors. Error status messages are routed depending on the error and whether valid optional destinations are available:

- If the prior-transaction request specifies a valid reply destination name (URBIRNAM), a status message is sent to the specified destination.
- If the prior-transaction request specifies a valid destination name (URBIDNAM), a status message is sent to the specified destination for certain errors (subcodes 34, 35, 37 and 39 to 43).
- If the reply destination name is also associated with the subscription, only one status message will be sent to that destination.
- If no active destination can be identified for an error status message, message ADAF1V will be generated on the operator's console and file DDPRINT.

Destinations receiving status messages must be for WebSphere MQ or webMethods EntireX messaging systems. Status messages contain the following information:

Element	Description
URBSTIME	STCK time when the **URBS** message was created.
URBSRTOK	User token provided with the **URBI**.
URBSSNAM	Subscription name provided with the **URBI**.
URBSDNAM	Destination name provided with the **URBI**.
URBSTSNR	Transaction sequence number within the subscription/destination provided with the **URBI**.
URBSRT	"TRAN"
URBSST	"TRSP"
URBSRSP	Response code.
URBSSUBC	Subcode.

This section covers the following topics:

- URBI Fields

- Prior-Transaction Request Example

URBI Fields

For a prior-transaction request, set the elements in the **URBI DSECT** as described in the following table. Values may be provided for only these fields. All unused binary fields must be set to zero, and all unused character fields must be set to blanks.

Element	Setting
URBIEYE	Set to "URBI".
URBILEN	Set to URBIL.
URBILENH	Set to URBIL.
URBIRT	Set to "TRAN".
URBISNAM	Supply the subscription name.
URBITSNR	Supply the transaction number for which retransmission is requested.
URBIDNAM	Supply the destination name associated with the supplied subscription for which retransmission is requested.
URBIRNAM	Optionally, supply a reply destination name.
URBIRTOK	Optionally, supply the eight-byte token that allows your target application to recognize different status requests.

Prior-Transaction Request Example

This example requests a prior-transaction run of the subscription named COLOR. The following topics are included in this section.

- Specify the replication parameters
- Create and send the prior-transaction request
- Review the output in the OUT1 and OUT2 destinations
- Review the output in the OUT3 destination

Specify the replication parameters

Assume that the RESEND1 resend buffer is defined with the size specified as shown below:

```
ADARPD RESENDBUFFER NAME=RESEND1
ADARPD RSIZE=2048K
```

Assume also that the COLOR subscription is associated with output destinations and the RESEND1 resend buffer as shown below:

```
ADARPD SUBSCRIPTION NAME=COLOR
ADARPD SFILE=4,SFDBID=10006,SFBAI='AA,AB.',SDESTINATION='OUT1,OUT2'
ADARPD SRESENDBUFFER=RESEND1
```

Create and send the prior-transaction request

The following application code requests retransmission of transaction 100 within subscription COLOR. The transaction will be sent to destinations OUT1 and OUT2, and a **URBS** status message will be sent to destination OUT3.

```
rr=newMessage(bb.send_buffer) # start a new message beginning with
    # URBH  and followed by URBI
rr.requestPriorTransaction(
      token=TOKTOK,
      snam=COLOR,
      tsnr=100,
      rnam=OUT3)

bb.send_length=rr.endMessage() # end message with
     # URBHLENT = total message size
     # and set length for messaging system

bb.send()       # send message
```

This is the resulting message with **URBH** and **URBI** settings generated from the status request that is sent to the Event Replicator input queue :

```
0000 E4D9C2C8 00000040 F0F10001 000000A0  URBH... 01......
0010 00000000 00000000 00000000 00000000  ................
0020 C1849489 95404040 00000000 00000000  Admin   ........
0030 00000000 00000000 00000000 00000000  ................
0040 E4D9C2C9 00000060 00000060 00000000  URBI............
0050 E3D6D2E3 D6D24040 D6E4E3F3 40404040  TOKTOK  OUT3
0060 E3D9C1D4 00000000 40404040 40404040  TRAN....
0070 C3D6D3D6 D9404040 40404040 40404040  COLOR
0080 00000000 00000000 00000000 00000000  ................
0090 00000000 00000000 00000000 00000000  ................
```

Review the output in the OUT1 and OUT2 destinations

If the transaction was found in the resend buffer, OUT1 and OUT2 will receive a message with a **URBH**, **URBT**, all **URBRs** and before-image **URBDs**, and ending with a **URBE**. The following fields are altered to indicate retransmission:

Element	Setting
URBTRSND	Set to "Y", indicating data may have been sent previously.
URBTPTRN	Set to "Y", indicating data from a previous transaction is being resent.
URBRRSND	Set to "Y", indicating data may have been sent previously.

If there were errors, OUT1 and OUT2 may each receive a URBS status message similar to the one described for **OUT3**.

Review the output in the OUT3 destination

The following depicts the received message at destination OUT3 for normal completion:

```
0000 E4D9C2C8 00000040 F0F10001 000000C0  URBH... 01......
0010 00000004 BB680AF2 C1C27F60 27170000  .......2AB"-....
0020 D9C5D7E3 D6D94040 00000000 00000000  REPTOR  ........
0030 00000000 00000000 00000000 00000000  ................
0040 E4D9C2E2 00000080 E3D6D2E3 D6D24040  URBS....TOKTOK
0050 E3D9C1D4 E3D9E2D7 BB680AF2 C1C18060  TRANTRSP...2AA.-
0060 00000000 00000000 40404040 40404040  ........
0070 40404040 40404040 C3D6D3D6 D9404040          COLOR
0080 40404040 40404040 BB680A7E 1AF84B02          ...=.8..
0090 00000000 00000000 00000064 00000000  ................
00A0 00000000 00000000 00000000 00000000  ................
00B0 00000000 00000000 00000000 00000000  ................
```

Status Requests

A status request inquires about the status of a subscription.

- If the status request specifies a subscription and destination, a status message about the subscription/destination pair is sent to the specified destination.
- If the status request specifies only the subscription, a status message about each subscription/destination pair in the subscription is sent to all destinations defined in the subscription.
- If the status request specifies only the destination, a status message about the subscription/destination pair is sent to the specified destination for each subscription having the destination as output.

In addition, a reply destination can be specified in the status request. The reply destination receives a copy of the status messages. Destinations receiving status messages must be for WebSphere MQ or webMethods EntireX messaging systems.

The status message (URBS) contains the following information:

Status Element	Description
URBSTTIM	STCK Time of transaction commit. In other words, when database processed the ET
URBSSTIM	STCK Time when last records was processed by subscription
URBSTIME	STCK Time when URBS message was sent
URBSTSNR	Transaction sequence number within subscription/destination

This section covers the following topics:

- URBI Fields
- Status Request Example

URBI Fields

For a status request, set the elements in the URBI DSECT as described in the following table.

Element	Setting
URBIEYE	Set to 'URBI'.
URBILEN	Set to URBIL.
URBILENH	Set to URBIL.
URBILEND	Set to 0.
URBIRT	Set to 'STAT'
URBISNAM	Supply the subscription name. Either the subscription name (URBISNAM) or the destination name (URBIDNAM) or both are required.
URBIDNAM	Supply the destination name. Either the subscription name (URBISNAM) or the destination name (URBIDNAM) or both are required.
URBIRNAM	Optionally supply a response destination.
URBIRTOK	Optionally supply the eight-byte token that allows your client program to recognize different status requests.

Status Request Example

This example requests a status of the subscription named COLOR.

- Specify the replication parameters
- Create and send the status request

- Review the output in the OUT1 destination

Specify the replication parameters

The COLOR subscription defines the format and output destination:

```
ADARPD SUBSCRIPTION NAME=COLOR,SFILE=4,SFDBID=10006,SFBAI='AA,AB.',SDESTINATION='OUT1'
```

Create and send the status request

The following application code inquires about the status of subscription COLOR.

```
rr=newMessage(bb.send_buffer)  # start a new message beginning with URBH

rr.requestStatus(   # followed by URBI
   token=TOKTOK,,
   snam=COLOR)

bb.send_length=rr.endMessage() # end message with URBHLENT = total message size
                               # and set length for messaging system

bb.send()                      # send message
```

This is the resulting message with URBH and URBI settings generated from the status request that is sent to the Event Replicator input queue :

```
    0000 E4D9C2C8 00000040 F0F10001 000000A0  URBH... 01......
    0010 00000000 00000000 00000000 00000000  ................
    0020 C1849489 95404040 00000000 00000000  Admin   ........
    0030 00000000 00000000 00000000 00000000  ................
    0040 E4D9C2C9 00000060 00000060 00000000  URBI...-...-....
    0050 E3D6D2E3 D6D24040 40404040 40404040  TOKTOK
    0060 E2E3C1E3 00000000 40404040 40404040  STAT....
    0070 C3D6D3D6 D9404040 40404040 40404040  COLOR
    0080 00000000 00000000 00000000 00000000  ................
         1 identical line(s) suppressed
```

Review the output in the OUT1 destination

The following depicts the received message on the OUT1 destination specified by the replication parameters:

```
    0000 E4D9C2C8 00000040 F0F10001 000000C0  URBH... 01......
    0010 00000004 BB680AF2 C1C27F60 27170000  .......2AB"-....
    0020 D9C5D7E3 D6D94040 00000000 00000000  REPTOR  ........
    0030 00000000 00000000 00000000 00000000  ................
    0040 E4D9C2E2 00000080 E3D6D2E3 D6D24040  URBS....TOKTOK
    0050 E2E3C1E3 E2E4C2E2 BB680AF2 C1C18060  STATSUBS...2AA.-
    0060 00000000 00000000 40404040 40404040  ........
    0070 40404040 40404040 C3D6D3D6 D9404040          COLOR
    0080 40404040 40404040 BB680A7E 1AF84B02          ...=.8..
    0090 BB680A7D F767EE02 00000002 00000000  ...'7...........
    00A0 00000000 00000000 00000000 00000000  ................
         1 identical line(s) suppressed
```

which can be interpreted as:

```
URBH element at offset X'0000'
     urbheye  = "URBH"
     urbhlen  = 64
     urbhvers = "01"
     urbhbord = X'0001'
     urbhlent = 192
     urbhmsnr = 4
     urbhtime = 2004-06-22 12:22:34.789927
     urbhrpid = 10007
     urbhrpni = 0
     urbhname = "REPTOR"

URBS element at offset X'0040'
     urbseye  = "URBS"
     urbslen  = 128
     urbsrtok = "TOKTOK"
     urbsrt   = "STAT"
     urbsst   = "SUBS" Subscription status information
     urbstime = 2004-06-22 12:22:34.789912
     urbsrsp  = 0
     urbssubc = 0
     urbserri = "        "
     urbsinam = "        "
     urbssnam = "COLOR"
     urbsdnam = "        "
     urbsptim = 2004-06-22 12:20:32.471940
     urbsttim = 2004-06-22 12:20:32.326270
     urbstsnr = 2
     urbsdbid = 0
     urbsfnr  = 0
```

6 Testing webMethods EntireX and WebSphere MQ Queues

- Using the ADAMTR (Message Test Receive) Utility 86
- Using the ADAMTS (Message Test Send) Utility 92

Event Replicator for Adabas provides two utilities you can use to test your EntireX and WebSphere MQ queues.

- The ADAMTR (Message Test Receive) utility can be used to receive one or more messages from an webMethods EntireX or WebSphere MQ queue, and, optionally, print the messages read from that queue.
- The ADAMTS (Message Test Send) utility can be used to send requests to the Event Replicator Server without having (or prior to writing) a target application.

Using the ADAMTR (Message Test Receive) Utility

The ADAMTR (Message Test Receive) utility can be used to receive one or more messages from an webMethods EntireX or WebSphere MQ queue and, optionally, print the messages read from that queue. This utility allows you to read and print messages sent to a queue by:

- an Event Replicator Server
- a target application
- the **ADAMTS utility**

The ADAMTR utility issues messages for any errors it encounters. For more information, read *ADAMTR Messages* in *Event Replicator for Adabas Messages and Codes Manual*.

This section covers the following topics:

- Syntax
- Parameters
- Running the Utility
- Stopping the Utility

Syntax

The overall syntax of the ADAMTR utility is:

```
ADAMTR  [MTYPE = { ETBROKER | MQSERIES }]
        [IQBUFLEN = { nnn [K] | 32767 ]
        [MSGLIMIT = nnn | 0 ]
        [PAGESIZE = nnnn | 55 ]
        [PRINT = { YES | NO ]
        { EntireX-Communicator-parameters | MQSeries-parameters }
```

The webMethods EntireX and WebSphere MQ parameters you can specify in ADAMTR are described in the following sections:

- webMethods EntireX ADAMTR Parameters
- WebSphere MQ ADAMTR Parameters

Note: Do not mix webMethods EntireX parameters and WebSphere MQ parameters in the same ADAMTR run.

webMethods EntireX ADAMTR Parameters

The parameters that can be used for the ADAMTR utility when testing an webMethods EntireX queue are shown in the following syntax. Note that all but the MTYPE and ETBSCONV parameter are required.

```
ADAMTR  [MTYPE = ETBROKER]
        [ETBBROKERNAME = { broker-name | BROKER }]
        ETBBROKERID = 'brokerid'
        ETBSERVICE = service
        ETBSERVICENAME = service-name
        ETBSERVICECLASS = service-class
        ETBSCONV = {YES | NO}
        ETBTOKEN = token-name
        ETBUSERID = user-id
```

Note: Do not specify any of the parameters associated with WebSphere MQ in the same ADAMTR utility run.

WebSphere MQ ADAMTR Parameters

The parameters that can be used for the ADAMTR utility when testing a WebSphere MQ queue are shown in the following syntax. Note that all parameters are required.

```
ADAMTR  MTYPE = MQSERIES
        MQQMGRNAME = MQ-manager-name
        MQQNAME = ' user.local.queue.name '
```

Note: Do not specify any of the parameters associated with webMethods EntireX in the same ADAMTR utility run.

Parameters

Parameters for the ADAMTR utility vary, depending on the kind of queue you are using. Each input parameter line to DDKARTE should begin with ADAMTR and is structured in the same manner as other Adabas utilities. All of the parameters that can be specified in the ADAMTR utility are now described.

MTYPE

This required parameter identifies the type of queue (webMethods EntireX or WebSphere MQ) that will be read by ADAMTR. Valid values are "ETBROKER" or "MQSERIES". The default for MTYPE is "ETBROKER".

ETBBROKERID

Use this parameter to specify the webMethods EntireX Broker ID to use for the ADAMTR run. This parameter is required when MTYPE=ETBROKER and should not be specified otherwise.

Broker IDs come in two formats: one for TCP/IP communications and one for Adabas SVC communications. For TCP/IP communications, the format is:

```
ip-address:port-number:TCP
```

In this case, the *ip-address* setting is the TCP/IP IP address and the *port-number* setting should match the webMethods EntireX PORT parameter.

For Adabas SVC communications, the format is:

```
'broker-id:SVCnnn:NET'
```

In this case, the *broker-id* setting should match the webMethods EntireX BROKER-ID parameter in the Broker ETBFILE DD. The *nnn* setting should match either the webMethods EntireX ADASVC or ADA5SVC parameters in the Broker PARMS DD statement.

ETBBROKERNAME

Use this parameter to specify the name of the EntireX Broker stub program to use for the ADAMTR run. This parameter can only be specified when MTYPE=ETBROKER. When not specified, this parameter defaults to "BROKER".

Notes:

1. EntireX Broker is a component of webMethods EntireX.

ETBSERVICE

Use this parameter to specify the webMethods EntireX service that should be used for the ADAMTR run. This should be the same as the value specified for the SERVICE parameter in webMethods EntireX. This parameter is required when MTYPE=ETBROKER and should not be specified otherwise.

ETBSERVICENAME

Use this parameter to specify the webMethods EntireX service name that should be used for the ADAMTR run. This should be the same as the value specified for the SERVER parameter

in webMethods EntireX. This parameter is required when MTYPE=ETBROKER and should not be specified otherwise.

ETBSERVICECLASS
Use this parameter to specify the webMethods EntireX service class that should be used for the ADAMTR run. This should be the same as the value specified for the CLASS parameter in webMethods EntireX. This parameter is required when MTYPE=ETBROKER and should not be specified otherwise.

ETBSCONV
Use this parameter to specify the webMethods EntireX queue is to run using single conversation mode. This parameter is optional when MTYPE=ETBROKER.

ETBTOKEN
Use this parameter to specify the webMethods EntireX queue is to run using single conversation mode. This parameter is optional when ETBSCONV=YES. When not specified, this parameter defaults to "ADAMTR".

ETBUSERID
Use this parameter to specify the webMethods EntireX queue is to run using single conversation mode. This parameter is optional when ETBSCONV=YES. When not specified, this parameter defaults to `REPTOR-<node>-----<jobname>`.

IQBUFLEN
Use this parameter to specify the length (in bytes) of the input buffer that will be allocated and used by the ADAMTR utility. The default for IQBUFLEN is 32,767 bytes. The minimum value is 2048 bytes and will be used if any lower value is entered for the IQBUFLEN parameter.

The theoretical maximum value that can be specified for this parameter is 2,147,483,647 bytes. In practice, however, the maximum value for IQBUFLEN specification is limited by the region size available to ADAMTR.

The value of the IQBUFLEN parameter should be set to a value at least as large as the largest expected message. For example, when reading messages written by an Event Replicator Server, a user may want to set IQBUFLEN to a value greater than or equal to the value set for the Event Replicator Server parameter MAXOUTPUTSIZE. If a message is read that is larger than the value specified for IQBUFLEN, the message will be truncated prior to its receipt by ADAMTR.

The suffix "K" may be used to denote that the value is in kilobytes (e.g. 1K = 1024 bytes, 4K = 4096 bytes, etc.).

MQQMGRNAME
Use this parameter to specify the WebSphere MQ queue manager name that should be used for the ADAMTR run. This parameter is required when MTYPE=MQSERIES and should not be specified otherwise.

MQQNAME

Use this parameter to specify the WebSphere MQ queue name that should be used for the ADAMTR run. This parameter is required when MTYPE=MQSERIES and should not be specified otherwise.

MSGLIMIT

Use this parameter to specify the number of messages processed by this ADAMTR utility run. Valid values range from zero (0) through 2,147,483,647. A value of zero indicates that there is no size limit for the number of messages processed by the utility. The utility will terminate when it has read and optionally printed the number of messages specified for MSGLIMIT. The default is zero.

Note: When a Broker queue is read by the ADAMTR utility, the utility must read input messaging transactions in the same manner the transactions were written to the queue. That is, assuming the sending program sends three messages and then commits the messaging transaction, ADAMTR will read the three messages and then commit the input messaging transaction. The value for MSGLIMIT is only checked by ADAMTR *after* committing an input messaging transaction. In the previously described transaction, if MSGLIMIT=1 or MSGLIMIT=2 is specified, ADAMTR will only terminate after reading and optionally printing the three messages.

PAGESIZE

Use this parameter to specify the number of lines to print prior to printing a new output page. The default value is 55 lines. The minimum value is 10 lines. If a value less than 10 is specified, the default value of 55 will be used.

PRINT

Use this parameter to indicate whether or not the ADAMTR utility should print each message it reads from the messaging system. Valid values are "YES" or "NO". The default is "YES". When PRINT=YES is specified, ADAMTR will print each message read from the messaging system. When PRINT=NO is specified, ADAMTR will not print the contents of each message. Use PRINT=YES when you want to see the content of each message; use PRINT=NO when you want to read messages from a queue but do not care or want to see the contents of each message.

Running the Utility

The JCL and input parameters for the ADAMTR utility are structured in the same manner as other Adabas utilities. Example JCL is shown below:

```
//(jobcard)
//*
//*
//* ADAMTR
//*
//*   Receive one or more messages from and Entire Broker or MQSeries
//*   Queue.
//*
//ADAMTR    EXEC PGM=ADARUN
//STEPLIB   DD DISP=SHR,DSN=USER.ADABAS.LOAD
//DDCARD    DD *
ADARUN PROGRAM=ADAMTR,SVC=xxx,DEVICE=dddd,DB=yyyyy
//DDDRUCK   DD SYSOUT=*
//DDKARTE   DD *
ADAMTR MTYPE=MQSERIES
ADAMTR MTYPE=ETBROKER
ADAMTR ETBBROKERNAME=BROKER
ADAMTR ETBBROKERID='III'
ADAMTR ETBSERVICE=SSSSSSSS
ADAMTR ETBSERVICENAME=SNSNSNSN
ADAMTR ETBSERVICECLASS=SCSCSCSC
ADAMTR MSGLIMIT=20000
ADAMTR PRINT=YES
//DDPRINT   DD SYSOUT=*
//SYSUDUMP  DD SYSOUT=*
```

Informational Messages

The following informational messages are displayed when ETBSCONV=YES.

```
Single Conversation Mode Set
Token  =
Userid =
```

Example showing default values

```
Single Conversation Mode Set
Token  = ADAMTR
Userid = REPTOR-DA3F-----USARCMTR
```

Stopping the Utility

You can stop ADAMTR utility execution in one of the following ways:

- If the MSGLIMIT parameter is set to a nonzero value, ADAMTR will terminate after processing the number of messages specified in the MSGLIMIT parameter.

  Note: When a Broker queue is read by the ADAMTR utility, the utility must read input messaging transactions in the same manner the transactions were written to the queue. That is, assuming the sending program sends three messages and then commits the messaging transaction, ADAMTR will read the three messages and then commit the input messaging transaction. The value for MSGLIMIT is only checked by ADAMTR *after* committing an input messaging transaction. In the previously described transaction, if MSGLIMIT=1 or MSGLIMIT=2 is specified, ADAMTR will only terminate after reading and optionally printing the three messages.

- ADAMTR will terminate after processing an input message that contains the eight-byte string "terminat" (all in lower case letters). With this in mind, a user could run ADAMTR with MSGLIMIT=0 (or MSGLIMIT set to a very high value) and then send this eight-byte string to the related queue using ADAMTS. When ADAMTR receives this special message, ADAMTR will optionally print the message and then terminate.

Using the ADAMTS (Message Test Send) Utility

The ADAMTS (Message Test Send) utility can be used to send requests to the Event Replicator Server without having (or prior to writing) a target application. The utility allows functional testing of:

- requests (for example, initial-state requests, prior-transaction requests, or status requests) sent to an Event Replicator Server via an webMethods EntireX or WebSphere MQ queue.
- an webMethods EntireX or WebSphere MQ queue, by sending one or more messages to that queue.

The ADAMTS utility issues messages for any errors it encounters. For more information, read *ADAMTS Messages* in *Event Replicator for Adabas Messages and Codes Manual*.

This section covers the following topics:

- Syntax
- Parameters

- Running the Utility

Syntax

The overall syntax of the ADAMTS utility is:

```
ADAMTS  [MTYPE = { ETBROKER | MQSERIES }]
        { EntireX-Communicator-parameters | MQSeries-parameters }
```

The webMethods EntireX and WebSphere MQ parameters you can specify in ADAMTS are described in the following sections:

- webMethods EntireX ADAMTS Parameters
- WebSphere MQ ADAMTS Parameters

Note: Do not mix webMethods EntireX parameters and WebSphere MQ parameters in the same ADAMTS run.

webMethods EntireX ADAMTS Parameters

The parameters that can be used for the ADAMTS utility when testing an webMethods EntireX queue are shown in the following syntax. Note that all but the MTYPE parameter are required.

```
ADAMTS  [MTYPE = ETBROKER]
        [ETBBROKERNAME = { broker-name | BROKER }]
        ETBBROKERID = 'brokerid '
        ETBSERVICE = service
        ETBSERVICENAME = service-name
        ETBSERVICECLASS = service-class
        ETBSCONV = {YES | NO}
        ETBTOKEN = token-name
        ETBUSERID = user-id
```

Note: Do not specify any of the parameters associated with WebSphere MQ in the same ADAMTS utility run.

WebSphere MQ ADAMTS Parameters

The parameters that can be used for the ADAMTS utility when testing a WebSphere MQ queue are shown in the following syntax. Note that all parameters are required.

```
ADAMTS  MTYPE = MQSERIES
        MQQMGRNAME = MQ-manager-name
        MQQNAME = ' user.local.queue.name '
```

Note: Do not specify any of the parameters associated with webMethods EntireX in the same ADAMTS utility run.

Parameters

Parameters for the ADAMTS utility vary, depending on the kind of queue you are using. Each input parameter line to DDKARTE should begin with ADAMTS and is structured in the same manner as other Adabas utilities. All of the parameters that can be specified in the ADAMTS utility are now described.

MTYPE

This required parameter identifies the type of queue (webMethods EntireX or WebSphere MQ) to which requests should be sent for the ADAMTS test. Valid values are "ETBROKER" or "MQSERIES". The default for MTYPE is "ETBROKER".

ETBBROKERID

Use this parameter to specify the webMethods EntireX Broker ID to use for the ADAMTS run. This parameter is required when MTYPE=ETBROKER and should not be specified otherwise.

Broker IDs come in two formats: one for TCP/IP communications and one for Adabas SVC communications. For TCP/IP communications, the format is:

```
ip-address:port-number:TCP
```

In this case, the *ip-address* setting is the TCP/IP IP address and the *port-number* setting should match the webMethods EntireX PORT parameter.

For Adabas SVC communications, the format is:

```
'broker-id:SVCnnn:NET'
```

In this case, the *broker-id* setting should match the webMethods EntireX BROKER-ID parameter in the Broker ETBFILE DD. The *nnn* setting should match either the webMethods EntireX ADASVC or ADA5SVC parameters in the Broker PARMS DD statement.

ETBBROKERNAME

Use this parameter to specify the name of the EntireX Broker stub program to use for the ADAMTS run. This parameter can only be specified when MTYPE=ETBROKER. When not specified, this parameter defaults to "BROKER".

Notes:

1. EntireX Broker is a component of webMethods EntireX.

ETBSERVICE

Use this parameter to specify the webMethods EntireX service that should be used for the ADAMTS run. This should be the same as the value specified for the SERVICE parameter in webMethods EntireX. This parameter is required when MTYPE=ETBROKER and should not be specified otherwise.

ETBSERVICENAME

Use this parameter to specify the webMethods EntireX service name that should be used for the ADAMTS run. This should be the same as the value specified for the SERVER parameter in webMethods EntireX. This parameter is required when MTYPE=ETBROKER and should not be specified otherwise.

ETBSERVICECLASS

Use this parameter to specify the webMethods EntireX service class that should be used for the ADAMTS run. This should be the same as the value specified for the CLASS parameter in webMethods EntireX. This parameter is required when MTYPE=ETBROKER and should not be specified otherwise.

ETBSCONV

Use this parameter to specify the webMethods EntireX queue is to run using single conversation mode. This parameter is optional when MTYPE=ETBROKER.

ETBTOKEN

Use this parameter to specify the webMethods EntireX queue is to run using single conversation mode. This parameter is optional when ETBSCONV=YES. When not specified, this parameter defaults to "ADAMTS".

ETBUSERID

Use this parameter to specify the webMethods EntireX queue is to run using single conversation mode. This parameter is optional when ETBSCONV=YES. When not specified, this parameter defaults to `REPTOR-<node>-----<jobname>`.

MQQMGRNAME

Use this parameter to specify the WebSphere MQ queue manager name that should be used for the ADAMTS run. This parameter is required when MTYPE=MQSERIES and should not be specified otherwise.

MQQNAME

Use this parameter to specify the WebSphere MQ queue name that should be used for the ADAMTS run. This parameter is required when MTYPE=MQSERIES and should not be specified otherwise.

Running the Utility

The ADAMTS utility reads messages from the MTSIN file in its JCL and sends those messages to an webMethods EntireX or WebSphere MQ queue. JCL and input parameters for ADAMTS are structured in the same manner as for other Adabas utilities. The MTSIN should be defined with RECFM=VB.

This section covers the following topics:

- Sample JCL
- Example of MTSIN Input Data

Sample JCL

The JCL and input parameters for the ADAMTS utility are structured in the same manner as other Adabas utilities. Example JCL is shown below:

```
//ADAMTS   EXEC PGM=ADARUN
//STEPLIB  DD DISP=SHR,DSN=USER.ADABAS.LOAD
//DDDRUCK  DD SYSOUT=X
//DDPRINT  DD SYSOUT=X
//SYSUDUMP DD SYSOUT=X
//DDCARD   DD *
ADARUN PROGRAM=ADAMTS,SVC=xxx,DEVICE=dddd,DB=yyyyy
/*
//DDKARTE  DD *
ADAMTS MTYPE=MQSERIES
ADAMTS MQQMGRNAME=MQnn
ADAMTS MQQNAME='user.local.queue'
/*
//*   The MTSIN file should be defined with RECFM=VB
//*   The messages to send are read from the MTSIN file.
//MTSIN    DD  DISP=SHR,DSN=ARFvrs.MVSMTSI
```

Informational Messages

The following informational messages are displayed when ETBSCONV=YES.

```
Single Conversation Mode Set
Token  =
Userid =
```

Example showing default values

```
Single Conversation Mode Set
Token  = ADAMTS
Userid = REPTOR-DA3F-----USARCMTS
```

This example JCL tests a WebSphere MQ queue.

Example of MTSIN Input Data

The file allocated by MTSIN in the JCL contains the messages to be read for the test. It should be defined with RECFM=VB.

The following sample MTSIN input data is shown with the hexadecimal offset in the first column, the hexadecimal representation of the data in the next four columns, and the actual data on the right of each line bounded by an asterisk.

```
0000 E4D9C2C8 00000040 F0F10001 000000A7 *URBH    01 .   x*
0010 00000000 00000000 00000000 00000000 *                 *
0020 C1D3C7D9 E3C8D4F1 00000000 00000000 *ALGRTHM1         *
0030 00000000 00000000 00000000 00000000 *                 *
0040 E4D9C2C9 00000067 00000060 00000007 *URBI   .   -   .*
0050 C1D3C7D9 E3C8D4F1 D6E4E3F1 40404040 *ALGRTHM1OUT1     *
0060 C9D5E2E3 00C7008F C9F1F9F9 F1F4F3F1 *INST G .I1991431*
0070 40404040 40404040 40404040 40404040 *                 *
0080 00000000 00000000 00000000 00000000 *                 *
0090 00000000 00000000 00000000 00000000 *                 *
00A0 C4C4C4C4 C4C4C4                     *DDDDDDD*
```

Index

A

ADAMTR utility, 86
  parameter descriptions, 88
  running, 90
  stopping, 92
  syntax, 86
  webMethods EntireX parameters, 87
  WebSphere MQ parameters, 87
ADAMTS utility, 92
  parameter descriptions, 94
  running, 96
  syntax, 93
  webMethods EntireX parameters, 93
  WebSphere MQ parameters, 94

C

client requests, 55
  destination open and close requests, 57
  initial-state requests, 60
  prior-transaction requests, 76
  status requests, 80
  URBH specifications, 56
closing destinations within an application, 57

D

destination open and close requests, 57
  example, 58
  URBI fields, 57
  URBS fields, 57
DSECTs
  URBC, 18, 20
  URBD, 19
  URBE, 20
  URBH, 21
  URBI, 22
  URBL, 25
  URBP, 37
  URBQ, 39
  URBR, 40
  URBS, 41
  URBT, 46
  URBU, 48
  URBX, 49
  URBZ, 51

E

ETBBROKERID parameter
  ADAMTR utility, 88
  ADAMTS utility, 94
ETBBROKERNAME parameter
  ADAMTR utility, 88
  ADAMTS utility, 95
ETBSCONV parameter
  ADAMTR utility, 89
  ADAMTS utility, 95
ETBSERVICE parameter
  ADAMTR utility, 88
  ADAMTS utility, 95
ETBSERVICECLASS parameter
  ADAMTR utility, 89
  ADAMTS utility, 95
ETBSERVICENAME parameter
  ADAMTR utility, 88
  ADAMTS utility, 95
ETBTOKEN parameter
  ADAMTR utility, 89
  ADAMTS utility, 95
ETBUSERID parameter
  ADAMTR utility, 89
  ADAMTS utility, 95
Event Replicator for Adabas
  programmer's reference, v
Event Replicator for Adabas client requests, 55
  destination open and close requests, 57
  initial-state requests, 60
  prior-transaction requests, 76
  status requests, 80
  URBH specifications, 56

I

initial-state requests, 60
  examples, 62
  ISN list format, 61
  responses, 62
  URBI fields, 60
IQBUFLEN parameter, 89
ISN list format, 61

M

MQQMGRNAME parameter

ADAMTR utility, 89
ADAMTS utility, 95
MQQNAME parameter
ADAMTR utility, 90
ADAMTS utility, 95
MSGLIMIT parameter, 90
MTYPE parameter
ADAMTR utility, 88
ADAMTS utility, 94

O

opening destinations within an application, 57
output message
detail, 17
sample, 9
summary, 5
time stamps, 6

P

PAGESIZE parameter, 90
PRINT parameter, 90
prior-transaction requests, 76
example, 78
URBI fields, 78
URBS fields, 77

Q

queues
testing webMethods EntireX and WebSphere MQ, 85

S

status requests, 80
example, 81
URBI fields, 81
URBS fields, 80

U

URBC DSECT, 18
URBD DSECT, 19
URBE DSECT, 20
URBH DSECT, 21
URBH specifications in client requests, 56
URBI DSECT, 22
URBL DSECT, 25
URBP DSECT, 37
URBQ DSECT, 39
URBR DSECT, 40
URBS DSECT, 41
URBT DSECT, 46
URBU DSECT, 48
URBX DSECT, 49
URBZ DSECT, 51
utilities
ADAMTR, 86
ADAMTS, 92

W

webMethods EntireX
ADAMTR parameters, 87
testing queues, 85
webMethods EntireX ADAMTS parameters, 93
WebSphere MQ
ADAMTR parameters, 87
testing queues, 85
WebSphere MQ ADAMTS parameters, 94