

Implementation of the Transport Service

This chapter describes, in its more detailed sections, the implementation of the transport service on host systems based on Natural and Adabas.

This chapter covers the following topics:

- Transport Service Terminology
 - Overview of the Transport Service Architecture
 - Usage by Con-nect
-

Transport Service Terminology

The following terminology is used with the transport service:

Transport Service address

Identifies senders (originators) and recipients of transport items transmitted by the transport service. A transport service address consists of two components, the node ID and application ID (see below).

Node ID

The identifier of a transport service node; it can be one to eight characters in length.

Application ID

The identifier of a transport service application; it can be one to eight characters in length. As a rule, the application ID corresponds to the Con-nect node type. See *Add a Mail Node*.

Lotus AU/Con-nect: Internet E-mail is an exception. It has application ID K and node type U.

Transport item

The unit of data distributed by the transport service. A transport item can be either a data object or status object (see below).

Data object

It contains:

- general information which identifies the item, its originating node and application;
- a list of one or more recipients which can include correlation data specific to each recipient of which the content and intended use is determined by the application;
- general correlation data of which the content and intended use is determined by the application; and
- user data.

Status object

It contains the same information as that in the data object, but with the following exceptions:

- the general information contains a reference to the data object along with a status code which indicates whether the referenced data object was successfully or not successfully delivered;
- it does not contain user data; and
- a status object itself has only one recipient, but it does contain the recipient and correlation data for each recipient of the data object.

Recipient

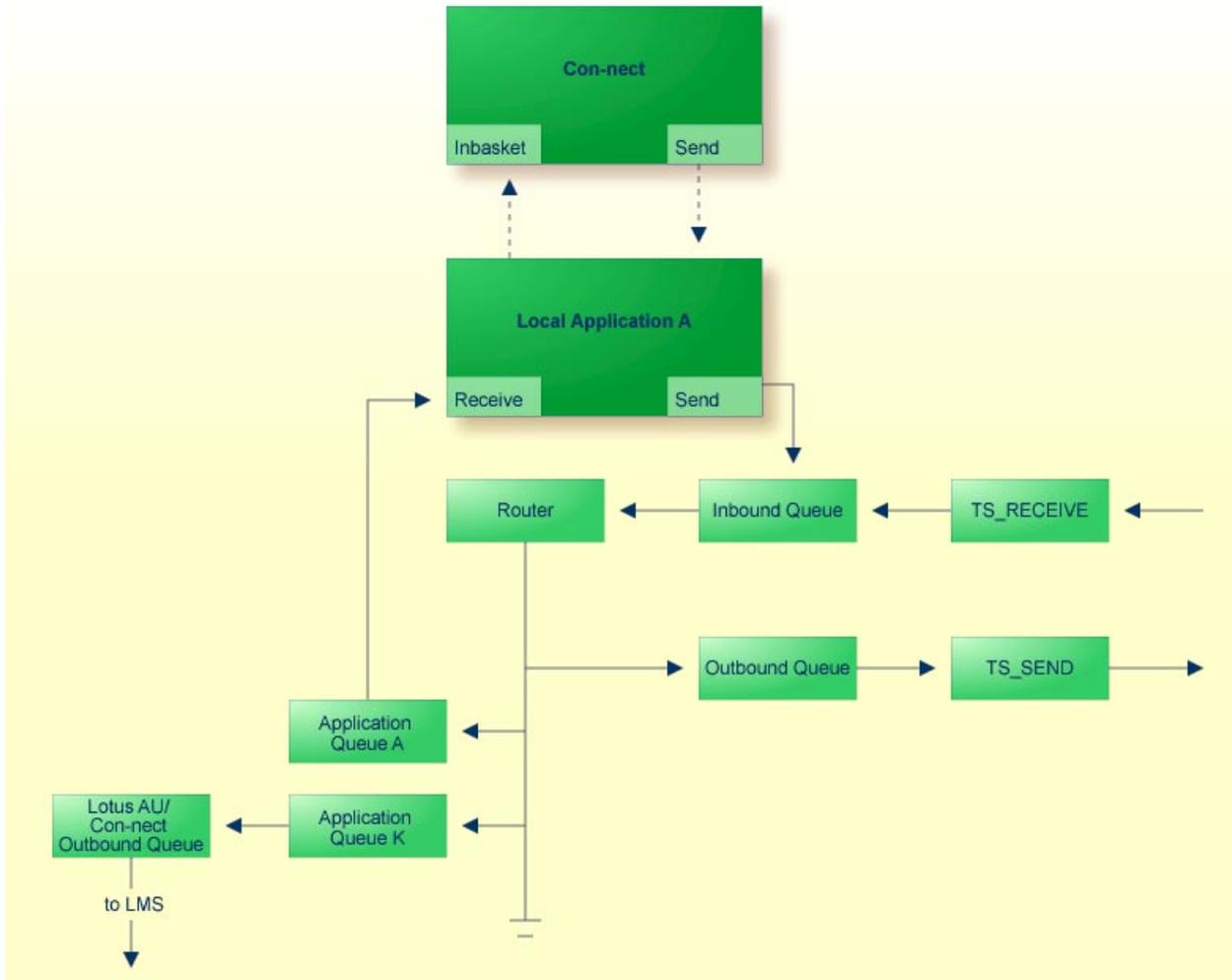
A transport service address. Transport items contain a list of one or more recipients to which the item is to be delivered. A recipient in this sense is a transport service application at a transport service node, and should not be confused with recipients specific to a particular electronic mail application using the transport service.

Feedback recipient

A transport service address to which status objects for a data object are to be delivered. This may differ from the originator's address.

Overview of the Transport Service Architecture

The following illustration provides an overview of the transport service.



Queues

The transport service maintains several types of queues in which transport items are stored between the various stages of processing.

Inbound Queue

The inbound queue is created when the transport service is initialized. There is only one inbound queue. Transport items received from other nodes as well as transport items created and submitted for delivery by the local transport items' applications are placed in this queue. They remain in the inbound queue until they are processed by the inbound queue server, TS_ROUTER, which routes the transport items to the local receiving applications, or to the appropriate outbound queues.

Outbound Queues

Outbound queues are created by the transport service administrator. The number of outbound queues required depends on the configuration of your transport service network. One or more of these queues must be created for each adjacent node in the transport service network with which, or by which, the transport service will communicate.

Transport items which are sent from or routed through the transport service are placed in the outbound queues where they remain until they are forwarded to the next node in their destination path by an outbound queue server process.

The outbound queues are typed according to the transport method used by the queue server.

OC62	A CICS LU6.2 queue.
OE62	An EntireX Broker Services (LU6.2 API and LU6.2 ACI) queue.
ORDA	A copy-to-other-database queue, or referred to in this documentation, as a remote database access (RDA) queue.

Application Queues

Application queues are created by the transport service administrator. They are used to accommodate local applications (e.g. Con-nect and Lotus AU/Con-nect). Objects sent to the local application are placed in these queues and remain there until they are processed by TS_ROUTER.

An application queue is type APPL.

Receiving Queues

Receiving queues are created by the transport service administrator and are required only when the local node uses Software AG's EntireX Broker Services (LU6.2 API or LU6.2 ACI) or remote database access. Receiving queues are used to control the demon processes which continuously listen for transaction requests from adjacent nodes and respond to the requests by executing the appropriate TS_RECEIVE program.

RE62	An EntireX Broker Services (LU6.2 API and LU6.2 ACI) queue.
RRDA	A remote database access (RDA) queue.

System Queues

Two system queues, type SYS, are automatically created when the transport service is initialized.

CR	A staging queue used by the transport service to temporarily store transport items during the creation process.
UD	This queue is used to hold undeliverable, or defective objects which you can examine at a later date.

Since there are no queue servers for the system queues no scheduling is required.

Queue Server Programs

Queue server programs are used to process and route the transport items through the various queues on to their destination.

Inbound Queue Server TS_ROUTER

Transport items in the inbound queue are processed by the queue server program, TS_ROUTER, which routes the items to various outbound queues or application queues based upon information given by the routing table (e.g. the local configuration and recipient information contained in the item). Routing is performed for each of the item's recipients as follows:

1. If the recipient is local, i.e. if an application queue is defined for the recipient, the transport item is copied to the appropriate application queue.
2. If the routing table contains a routing entry for the recipient node, the value in the "Next Destination Queue" field on the Routing Entry screen determines which outbound queue is used to forward the item closer to its destination and the item is copied to this queue.
3. If the recipient does not satisfy either of the previous conditions, a routing error occurs and, depending on the feedback options specified in the item, a status notification is created and returned to the sender.

Since a transport item can have more than one recipient, it can be routed to more than one application and outbound queue. In such situations, known as "fan-out", the router creates as many copies of the item as needed and marks each recipient's server of the respective queue with a "responsibility flag" in the item.

For a description of how the router program is scheduled, see *Scheduling*.

Outbound Queue Servers TS_SEND

The outbound queue servers' program, TS_SEND, processes the items in the outbound queues, forwarding them to other transport service nodes by means of the transport method determined by the queues. The queue servers convert the items to a standard transmission format and then perform the actual transport to the other nodes.

- LU6.2 (CICS) for queue type OC62 (the TS_SEND server of an LU6.2 (CICS) queue can only run as a started task under CICS);
- LU6.2 (EntireX) for queue type OE62;
- Remote database access for queue type ORDA.

TS_SEND programs exist for each outbound queue type. For a description of how they are scheduled, see *Scheduling*.

External Receive Programs TS_RECEIVE

Transport items sent from other nodes are received by different server programs depending upon the transport method involved and corresponding to the type of TS_SEND program. They are responsible for receiving the transport items, converting them from a transmission format to an internal format and placing them in the inbound queue for processing by the TS_ROUTER.

Depending upon the transmission method used by the corresponding remote TS_SEND program, different implementations of TS_RECEIVE are available:

TS_RECEIVE LU6.2 (CICS)

Accepts transport items from TS_SEND LU6.2 programs (CICS or EntireX Broker Services (LU6.2 API or LU6.2 ACI)). With this implementation, TS_RECEIVE runs as a started-task under CICS.

TS_RECEIVE LU6.2 (EntireX Broker Services (LU6.2 API))

Accepts transport items from TS_SEND LU6.2 programs (CICS or EntireX Broker Services (LU6.2 API)).

TS_RECEIVE LU6.2 (EntireX Broker Services (LU6.2 ACI))

Accepts transport items from TS_SEND LU6.2 programs (CICS or EntireX Broker Services (LU6.2 ACI)).

TS_RECEIVE RDA

Accepts transport items from TS_SEND RDA programs (i.e. locally copying and re-formatting transport items from an intermediate queue into the inbound queue).

Application Programs

Applications which utilize the transport service interact with it in two different scenarios:

- as senders which create transport items and submit them to the transport service for delivery. These items are placed in the inbound queue where they are processed by TS_ROUTER.
- as recipients which receive transport items delivered to them by the transport service from other applications. These items are queued for the application in the corresponding application queue by TS_ROUTER.

Scheduling

The transport service provides the following modes of scheduling for the queue servers:

Event Driven Scheduling

the server is activated whenever an item arrives in the queue.

Timer Driven Scheduling

the server is activated at the time interval you specify.

The scheduling modes are used in connection with the following queue statuses: hold, inactive, event driven, and timer driven. As the transport service administrator, you must ensure that the appropriate mechanisms for scheduling the transport service queue server programs are established and activated. The kind of mechanism to be applied depends upon the output status or reset status of the respective queue.

The output status indicates the current status of the queue processing, whereas the reset status reflects the general type of scheduling mechanism to be applied to the respective queue.

Status	Description
Hold	The server process is not scheduled nor activated until this status is changed.
Inactive	The server process is only executed when you explicitly activate it.
Event Driven	The server process is activated each time an entry is added to that queue.
Timer Driven	The server process re-schedules itself each time it is about to terminate its execution. The initial execution must be activated by some other mechanism (e.g. activated by the administrator).
Active	Only for output status - the server process is currently active.
Scheduled	Only for output status - the server process has been scheduled. In the case of timer driven scheduling, that status persists during the specified execution.

The scheduler keeps track of the last time each queue was scheduled and when the respective queue last started and stopped its execution. You can monitor the "time stamps" from the "Queue Maintenance" screen. See *Maintaining the Transport Service*.

In contrast to other queue types, the receiving queues are subject to timer driven scheduling regardless of whether their status is set to inactive, event driven or timer driven. The delay time for re-execution is fixed to one minute.

Scheduling mechanisms as described here, and their control by means of administration functions, are available only when queue servers are run in environments such as Com-plete where the servers run as attached tasks, or CICS where the servers run as started-tasks.

If the queue servers are run in batch mode, a simplified method must be applied whereby at least two Natural sessions are used by a general server to poll the queues in a regular and serialized manner.

In the case of Com-plete or CICS, a task re-initiation program (i.e. watchdog) can be used to ensure that regular scheduling is not interrupted by temporary system failures.

The transport service administrator should occasionally check the queues' status, the number of entries stored in each of the queues and the time stamps to monitor the state of the system.

Usage by Con-nect

The transport service is used to transmit electronic mail between different types of messaging facilities. Typically, the external mail facilities are Con-nect itself and gateways to third-party systems. Therefore the transport service allows a single gateway to be used by more than one Con-nect system and allows it to be located on the same or a platform other than that used by Con-nect .

When the transport service is used to support electronic mail services, a two-layered addressing scheme is used. The first- or outer-layer of the address information is evaluated by the transport service when transmitting mail items and notifications between different external mail facilities. The second- or inner-layer of the address information is evaluated by the external mail facilities which use the transport service.

For example, the second-layer address may be a Con-nect node and cabinet ID. Even though the transport service does not enforce restrictions on how the second-layer addressing works, or even how the addressees are structured, it supports linkage between the two layers of the address information so that status information can be evaluated by both the transport service and the external mail facilities serviced.

Naming and addressing in the transport service has been reduced to the most basic components - to nodes and entities within the nodes called applications. The format of the IDs for both nodes and applications is that of a printable character string, which can range between one and eight characters in length. If the transport service is to be used as an electronic mail backbone facility for Con-nect gateways, then the length of the application ID must be exactly one character.

It is important to understand that the transport service addresses, i.e. the node and application IDs, are independent of the names used in the underlying network services (e.g. SNA LU names, and OSI SSAP addresses) and of names in the electronic mail facilities which are serviced (e.g. Con-nect user names, Internet e-mail addresses).

The routing of data units, so called transport items, is done on the basis of the node names, which thereby localize the point where the respective service is provided.

In the electronic mail application of the transport service, the application ID identifies the type of mail facility. For example, A signifies a Con-nect external mail node. Exception: the application ID K signifies a Lotus AU/Con-nect: Internet e-mail node (node type U).

This means, for example, when a mail item is transmitted from a Con-nect system to the Lotus AU/Con-nect interface, the transport service will transmit a transport item from application A on one node to application K on the same node.

Note:

The transport service allows for multi-destination distribution. Thus, a mail item can be transmitted from Con-nect to the external mail facility of another Con-nect system, as well as to the Lotus AU/Con-nect interface, all at the same time. In addition, dividing the transport service addresses into two components - node and application - means that the gateways for multiple electronic mail facilities can be set up on one and the same, or on different systems.