

Implementation of Con-nect SNADS

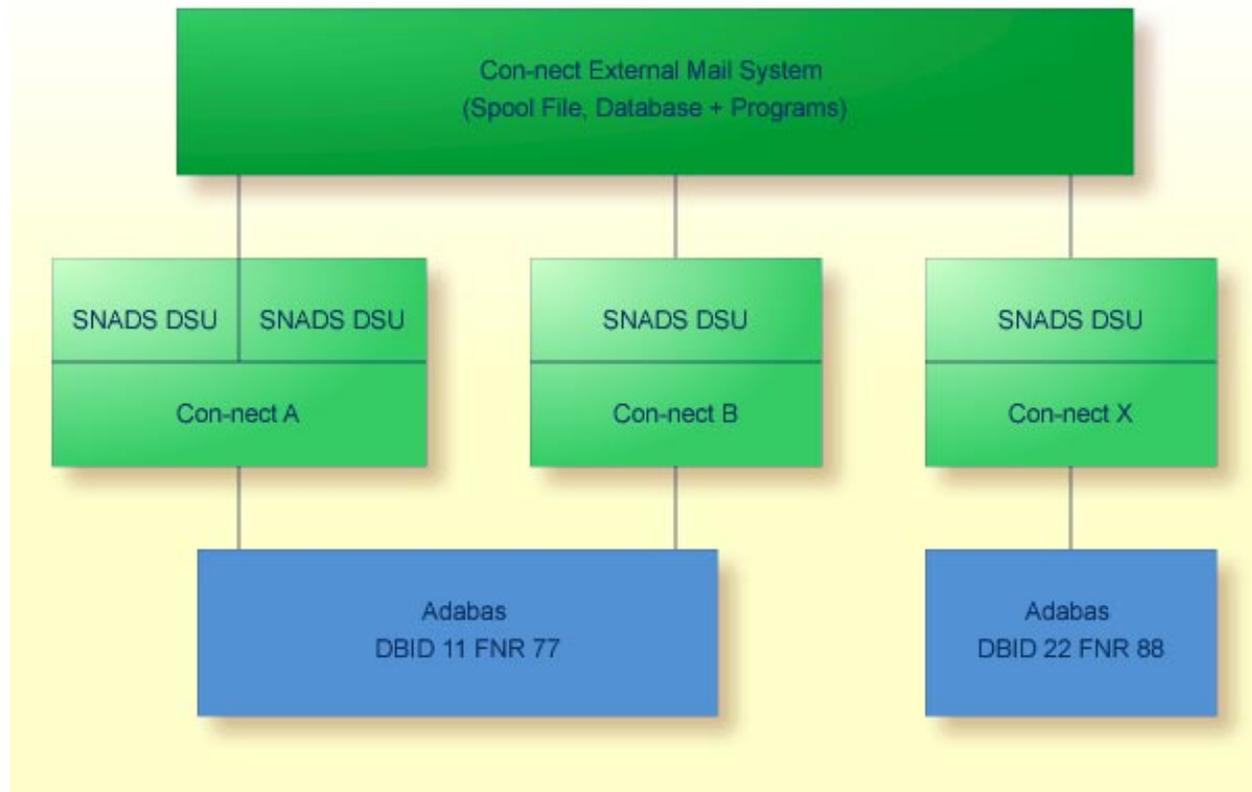
Con-nect SNADS consists of data structures in the Con-nect spool file as well as programs which implement SNADS functions and provide the interface between SNADS and the Con-nect Send function. This section provides an overview of the implementation of Con-nect SNADS.

This chapter covers the following topics:

- Con-nect as a SNADS Node
 - Con-nect SNADS Queues
 - Con-nect SNADS Server Processes
 - Con-nect SNADS Routing Table
 - Scheduling the Con-nect SNADS Queue Servers
-

Con-nect as a SNADS Node

Con-nect SNADS is one, of possibly several, external mail types which share a Con-nect spool file. A single spool file or Con-nect external mail system can serve multiple Con-nect systems. Each Con-nect system can function as multiple SNADS DSUs. The following diagram illustrates these relationships.



Con-nect SNADS Queues

Con-nect SNADS maintains three types of queues in which SNADS DIUs are stored between the various stages of processing.

Dummy Queue

This queue is created when Con-nect SNADS is initialized with systems using Software AG's EntireX Broker Services (LU6.2 API). There is only one dummy queue. It is used to control the demon process. The demon process is used to continuously listen for conversation requests from remote SNADS partner nodes and to execute the DS_RECEIVE program when a request is caught.

Inbound Queue

This queue is created when Con-nect SNADS is initialized. There is only one inbound queue. SNADS DIUs received from other nodes and SNADS DIUs sent by Con-nect users are immediately placed in this queue, where they remain until they are processed by the inbound queue server, DS_ROUTER_DIRECTOR, which either directs the distributions to local Con-nect users, or routes them to the appropriate outbound queue.

Outbound Queues

You create these queues when configuring Con-nect SNADS. The number of outbound queues depends on the configuration of your particular SNADS network. One or more of these queues must be created for each adjacent node in the SNADS network with which, or via which, Con-nect SNADS will communicate.

SNADS DIUs which are sent from or routed through Con-nect SNADS are placed in these queues and remain there until they are forwarded on to the next node by the outbound queue server process DS_SEND.

Con-nect SNADS Server Processes

SNADS DIUs are processed through the queues described in the previous section. The DIUs are processed by queue servers and other server processes.

The Con-nect SNADS queue servers are DS_ROUTER_DIRECTOR and DS_SEND.

DS_RECEIVE is another server process, however, it does not take the DIUs which are to be processed from a queue. Instead, it takes the DIUs from the DS_SEND process of an adjacent SNADS node - in the context of an APPC conversation which is based on the SNA LU6.2 protocol.

These server processes function differently within the three possible SNADS environments. Therefore, a separate description of how the server processes function with Com-plete and EntireX Broker Services (LU6.2 API), in batch mode and EntireX Broker Services (LU6.2 API), and with CICS is given below.

Com-plete and EntireX Broker Services (LU6.2 API)

DS_RECEIVE (Inbound queue: non-scheduled)

The DS_RECEIVE program processes DIUs which are received from an adjacent node and inserts them into the inbound queue of the current node. A Con-nect SNADS "demon" process continuously listens for conversation requests from remote SNADS partner nodes. When a request is caught, the demon process executes the DS_RECEIVE program as an inline subroutine.

The demon process itself runs as an attached task in the Com-plete environment and uses a dedicated Con-nect SNADS queue called the dummy queue for task control purposes.

The Con-nect SNADS supplied startup routine, CPSEND, leads the flow of control into the appropriate Natural environment.

DS_ROUTER_DIRECTOR (Inbound queue: scheduled)

If the recipients of the DIUs are elements (addressees) of a remote SNADS node, the "router" component of the DS_ROUTER_DIRECTOR program processes the DIUs from the inbound queue into one or several of the local system's outbound queues.

Whereas, if the recipients of the DIUs are elements (addressees) of the local Con-nect system, the "director" component of the DS_ROUTER_DIRECTOR program processes the DIUs from the inbound queue into the appropriate multi-node routine. This routine delivers the DIUs into the recipients' Inbasket.

DS_ROUTER_DIRECTOR is invoked by Con-nect SNADS with the Com-plete ATTACH function. The Con-nect SNADS supplied startup routine, CPSEND, leads the flow of control into the appropriate Natural environment.

DS_SEND (Outbound queues: scheduled)

The DS_SEND program processes DIUs from an outbound queue of the current node into the DS_RECEIVE program of an adjacent node.

Con-nect SNADS invokes the DS_SEND process with the Com-plete ATTACH function. The Con-nect SNADS supplied startup routine, CPSEND, leads the flow of control into the appropriate Natural environment.

Batch Mode and EntireX Broker Services (LU6.2 API)

DS_RECEIVE

The DS_RECEIVE program processes DIUs which are received from an adjacent node and inserts them into the inbound queue of the current node. A Con-nect SNADS "demon" process continuously listens for conversation requests from remote SNADS partner nodes. When a request is caught, the demon process executes the DS_RECEIVE program as an inline subroutine.

The demon process itself runs in batch mode where it exclusively occupies a continuously executing Natural task called the "input handler" and uses a dedicated Con-nect SNADS queue called the dummy queue for task control purposes.

DS_ROUTER_DIRECTOR

If the recipients of the DIUs are elements (addressees) of a remote SNADS node, the "router" component of the DS_ROUTER_DIRECTOR program processes the DIUs from the inbound queue into one or several of the local system's outbound queues.

Whereas, if the recipients of the DIUs are elements (addressees) of the local Con-nect system, the "director" component of the DS_ROUTER_DIRECTOR program processes the DIUs from the inbound queue into the appropriate multi-node routine. This routine delivers the DIUs into the recipients' Inbasket.

DS_ROUTER_DIRECTOR is executed by Con-nect SNADS with a continuously running Natural task called the "queue server". The DS_ROUTER_DIRECTOR can share this batch task with the DS_SEND program of one or multiple outbound queues but not with the demon process.

DS_SEND

The DS SEND program processes DIUs from an outbound queue of the current node into the DS_RECEIVE program an adjacent node.

Con-nect SNADS executes the DS_SEND process in a continuously running Natural task called the "queue server". The DS_SEND can share this batch task with the DS_ROUTER_DIRECTOR program of one or multiple outbound queues but not with the demon process.

CICS

DS_RECEIVE (Inbound queue: non-scheduled)

The DS RECEIVE program processes DIUs which are received from an adjacent node and inserts them into the inbound queue of the current node. DS_RECEIVE is invoked by CICS whenever an incoming conversation request is received from a remote partner. The Con-nect SNADS supplied startup routine, CSRECV, leads the flow of control into the appropriate Natural environment.

DS_ROUTER_DIRECTOR (Inbound queue: scheduled)

If the recipients of the DIUs are elements (addressees) of a remote SNADS node, the "router" component of the DS_ROUTER_DIRECTOR program processes the DIUs from the inbound queue into one or several of the local system's outbound queues.

Whereas, if the recipients of the DIUs are elements (addressees) of the local Con-nect system, the "director" component of the DS_ROUTER_DIRECTOR program processes the DIUs from the inbound queue into the appropriate multi-node routine. This routine delivers the DIUs into the recipients' Inbasket.

Con-nect SNADS invokes the DS_ROUTER_DIRECTOR program by means of the CICS interval control START command. The Con-nect SNADS supplied startup routine, CSSEND, leads the flow of control into the appropriate Natural environment.

DS_SEND (Outbound queues: scheduled)

The DS_SEND program processes DIUs from an outbound queue of the current node into the DS_RECEIVE program of an adjacent node.

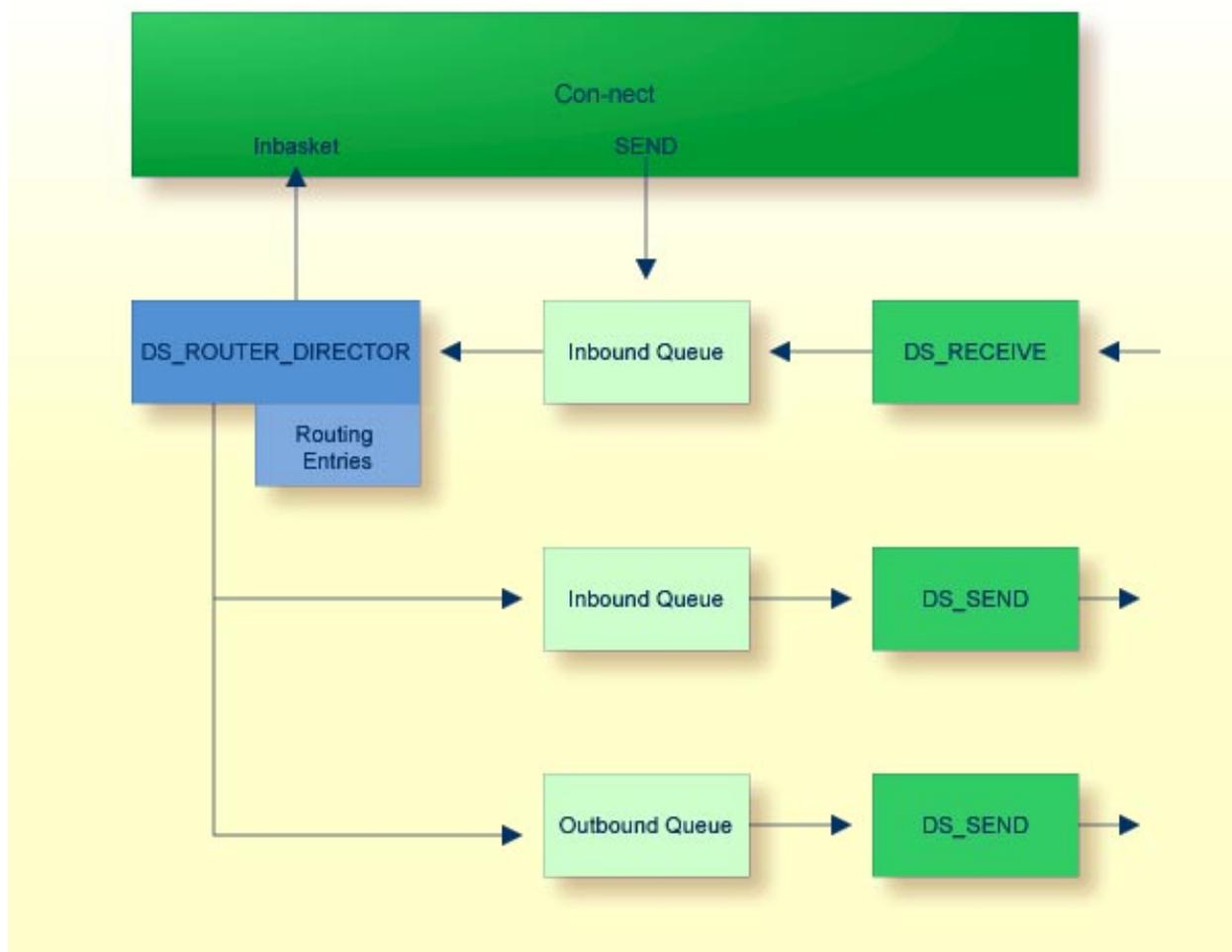
Con-nect SNADS invokes the DS_SEND process by means of the CICS interval control START command. The Con-nect SNADS supplied startup routine, CSSEND, leads the flow of control into the appropriate Natural environment.

Con-nect SNADS Routing Table

Certain information concerning the SNADS network must be available to Con-nect in the form of a routing table. The routing table specifications reside in the Con-nect spool file.

The routing table contains routing entries which enable the DS_ROUTER_DIRECTOR to determine the adjacent SNADS node to which the DIU should be transmitted (i.e. to which of the Con-nect SNADS queues the DIU should be appended). If an appropriate node does not exist, the DS_ROUTER_DIRECTOR can create another SNADS DIU which is to notify the originator of the DIU of the routing error. This depends on the feedback options specified in the DIU.

The following diagram shows the relationship of queues, routing entries and server programs:



Scheduling the Con-nect SNADS Queue Servers

Con-nect SNADS 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 time intervals specified by the administrator.

These scheduling modes are used in connection with the following queue statuses: hold, inactive, event driven, and timer driven. You must ensure that the appropriate mechanisms for scheduling the Con-nect SNADS queue server programs have been established and activated. The kind of mechanism to be applied depends upon the output status (or reset status) of 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 explicitly activated by an administrator.
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 have been activated by some other mechanism (e.g. activated by an administrator).

Note:

Task scheduling cannot be used in an EntireX Broker Services (LU6.2 API) batch mode configuration.

As the Con-nect SNADS administrator, you should occasionally check the queue statuses, the number of entries stored in each of those queues and their time stamps to monitor the state of the system.