

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION.....	1
1.1 Advantages.....	1
1.1.1 Security.....	2
1.1.2 Performance.....	2
1.1.3 Load Balancing.....	2
1.1.4 Flexibility.....	2
1.2 Basic Operation.....	3
CHAPTER 2: FEATURES.....	5
2.1 Standard Handling.....	5
2.2 PERMIT/REJECT Session.....	5
2.3 PERMIT/REJECT Crossing Traffic.....	6
2.4 Adjacent SSCP Selection with Respect to LUs.....	6
2.5 Adjacent SSCP Selection with Respect to VTAM.....	6
2.6 Alias Functions.....	6
2.7 Gateway Path Selection with Respect to LUs.....	6
2.8 Authorization Time Frames.....	7
2.9 SMF Recording.....	7
2.10 Reverse Mode.....	7
2.11 Session Management Definitions.....	7
2.11.1 Using the ISPF Interface.....	7
2.11.2 Coding the Macros.....	7
2.11.3 Using Generic Names.....	8
2.12 Dynamic Functions.....	8
2.13 User Exits.....	8
2.14 Testing Facilities.....	9
2.14.1 Simulator.....	9
2.14.2 Warning Mode.....	9
CHAPTER 3: COMPONENTS.....	11
3.1 Load Module ISTECAAA.....	11
3.2 Load Module SMONTAB.....	11
3.3 Control Table.....	11
3.4 ISPF Interface.....	12
3.5 Load Module ISTECSIM.....	12
CHAPTER 4: INSTALLATION AND OPERATION.....	13
4.1 Installation.....	13
4.2 Maintaining the Tables.....	13
4.3 Operation.....	13

CHAPTER 1: INTRODUCTION

Pilot for Sessions is a VTAM Session Management Exit (SME) program that allows you to control SNA sessions. By improving the functionality of VTAM at the session level, *Pilot for Sessions* can help you to achieve greater flexibility, security, and adaptability from your multiple domain or interconnected SNA networks.

Using *Pilot for Sessions* in every kind of SNA network is recommended. However, special importance is realized in multiple domain networks or SNA interconnected (SNI) networks. In general, a single domain network allows you to control the entire network completely. This is no longer possible in larger and meshed networks, especially those who are connected to other companies. In this case, it is not always known which SNA networks belong to the entire SNI network. The different SNA domains are no longer self-sufficient. They are points in a more global network which is no longer controlled centrally. Therefore, each subnet should define its own boundaries in such a way that only known and desired data traffic passes through them. *Pilot for Sessions* can implement this necessary control at the SNA session level. It is not enough to capture the data traffic occurring in the network, e.g. by a network management system. Such recordings have to be evaluated and they can show gaps in the SNA net boundaries only afterwards. The best alternative is to decide on the permission or rejection of the session establishment at the time when a specific session request occurs.

Pilot for Sessions eliminates the compromises that are ordinarily involved with SNA session management. For example, if you use only the standard VTAM functionality on a cross-domain network, you can improve security by defining all cross-domain resources explicitly. However, the price for this improved security includes increased complexity. With *Pilot for Sessions*, on the other hand, you can create significantly fewer explicit definitions by using generic names. *Pilot for Sessions* establishes control at the session level with fewer definitions and without relinquishing the use of VTAM's dynamic cross-domain resources.

1.1 Advantages

Pilot for Sessions offers significant advantages:

- Security
- Performance
- Flexibility
- Session management at the SNA level
- Load balancing

These advantages are detailed in the following paragraphs.

1.1.1 Security

In addition to checking the authorization of both session partners, *Pilot for Sessions* manages the session itself. *Pilot for Sessions* stops unauthorized access before the session even reaches the application.

During its authorization check, *Pilot for Sessions* uses the VTAM names of the session partners. You can use *Pilot for Sessions* for test installations of applications for restricted terminal groups.

Pilot for Sessions allows you to control the "crossing traffic" that passes through your network as a result of cross-domain sessions. In addition, *Pilot for Sessions* can help you to balance the load in the SNI network. *Pilot for Sessions* can also help you to manage international networks by taking into account the different costs for the links.

1.1.2 Performance

Pilot for Sessions modifies the adjacent SSCP selection of VTAM by assigning adjacent SSCPs to individual LUs or groups of LUs. This assignment allows *Pilot for Sessions* to control and accelerate the routing of LU transmissions through the network. *Pilot for Sessions* manages these assignments by using the user's specifications to modify VTAM's adjacent SSCP table. *Pilot for Sessions* keeps the number of assignments to a minimum by using generic names whenever possible. Additional cross-domain resource definitions for such LUs are not necessary.

The definitions in the *Pilot for Sessions* adjacent SSCP table can also reduce unnecessary searching for unknown resources in VTAM's own domain.

1.1.3 Load Balancing

Pilot for Sessions can modify VTAM's gateway path selection for specific LUs or groups of LUs. The assignment of a desired gateway NCP to specific LUs allows you to separate and control the traffic passing through your gateway NCPs on the session level. *Pilot for Sessions* handles the gateway path selection similar to the adjacent SSCP selection.

1.1.4 Flexibility

Pilot for Sessions offers a high degree of flexibility in defining sessions to be authorized. Flexibility advantages for session definitions include:

- Generic notation can be used in any definition.
- Sessions may be defined in more detail by specifying the session origin or requester.
- Session authorization may be limited to specific time frames.
- Reverse mode definition for one or both session partners (see paragraph 2.10 on page 7).

1.2 Basic Operation

VTAM handles *Pilot for Sessions* as an internal subroutine. *Pilot for Sessions* is initialized during VTAM startup and terminates when VTAM terminates. With VTAM version 3.4.1 or higher, *Pilot for Sessions* may also be initialized or terminated by issuing the “MODIFY NET,EXIT” command.

VTAM calls *Pilot for Sessions* whenever a session request occurs between two logical units (LUs). *Pilot for Sessions* compares the parameters of both session partners with a set of internal tables and uses this comparison to decide whether to continue establishing the session. *Pilot for Sessions* then records this activity to SMF. When the network is running, *Pilot for Sessions* is called in each of the following situations:

- At each session request, as soon as the destination logical unit (LU) for the session is known.
- At each session request that requires cross-domain searches of VTAM.
- At each session request which requires alias name translation.
- At each session request which requires gateway path selection.

Pilot for Sessions maintains a control table that stores your specifications for handling the establishment of SNA sessions. The specific session handling features offered by *Pilot for Sessions* are discussed in the next chapter.

Pilot for Sessions gains control of each SNA logon request and carries out validity checks. As a result, *Pilot for Sessions* can reject or accept any session request. You can also use the authorization testing function to monitor sessions that cause crossing traffic through a VTAM node on which *Pilot for Sessions* resides.

CHAPTER 2: FEATURES

Pilot for Sessions offers the following general features:

- Authorizing single-domain and multiple-domain SNA sessions between an application and the LUs that use the application.
- Implementing SMF accounting for each session start, end, and reject.
- Allowing you to translate aliases directly in VTAM, without using NetView or a similar program.
- Adapting the adjacent SSCP selection (if necessary) to the respective session.
- Adapting the gateway path selection (if necessary) to the respective session.
- Dynamic functions.
- Convenient testing facilities.
- User exit facilities.

The following sections explain these features in greater detail.

2.1 Standard Handling

The standard handling feature determines the default handling of session requests. If the default handling is defined as REJECT, then session establishment is granted only for the sessions that you have defined specifically with a PERMIT. If the default handling is defined as PERMIT, then you define only specific sessions that you want to reject.

2.2 PERMIT/REJECT Session

For important terminals and applications, you can create separate permissions for all sessions in the network. Before session establishment, *Pilot for Sessions* checks each session request and compares that request to an internal set of authorization definitions. *Pilot for Sessions* returns the result of this test to VTAM. Depending on the parameter that is generated (PERMIT or REJECT), VTAM continues or stops establishing the session.

2.3 PERMIT/REJECT Crossing Traffic

Pilot for Sessions checks each crossing session request to see if the crossing traffic is allowed or forbidden. The procedure used to check for crossing traffic is similar to the procedure that *Pilot for Sessions* uses for checking sessions which end in its own VTAM domain.

Note: As a prerequisite for this feature, VTAM must be an active participant in session establishment (for example, Gateway VTAM).

2.4 Adjacent SSCP Selection with Respect to LUs

For LUs that VTAM does not find within its own domain and LUs that request session partners outside their own domains, *Pilot for Sessions* will look internally for routing information. This check determines whether a specific adjacent SSCP table should be used for the LU. If so, *Pilot for Sessions* substitutes the VTAM adjacent table for the specific LU.

2.5 Adjacent SSCP Selection with Respect to VTAM

With *Pilot for Sessions*, cross-domain requests that cannot be satisfied in an adjacent VTAM can be routed directly to a gateway VTAM in the network. *Pilot for Sessions* tests for this condition by searching through its own definitions for a list of adjacent SSCPs specific to the destination VTAM. This table (together with the SSCP table described in paragraph 3.1.4, "Adjacent SSCP Selection with Respect to LUs,") also substitutes the standard adjacent SSCP table of VTAM.

2.6 Alias Functions

If necessary, *Pilot for Sessions* takes over the alias translation of LU names, Logmode, and COS names. This feature integrates the alias translation directly into VTAM, because *Pilot for Sessions* runs as a VTAM subroutine. If *Pilot for Sessions* is installed in different gateway VTAMs in an SNI network, each VTAM can maintain its own alias translation tables.

2.7 Gateway Path Selection with Respect to LUs

The gateway path selection feature of *Pilot for Sessions* is an optional feature available on request. With this feature you are allowed to route sessions over specific NCPs. *Pilot for Sessions* maintains user defined assignments of LUs and gateway NCPs and uses them to replace VTAM's standard gateway path selection list. The gateway NCPs can be defined either by name or by subarea number.

2.8 Authorization Time Frames

The validity of a *Pilot for Sessions* authorization definition can be restricted to a user defined time frame. For example, it is possible to permit sessions during the day time and reject those same sessions during the night.

2.9 SMF Recording

Pilot for Sessions can write SMF records for each session. When session recording is active, *Pilot for Sessions* writes a separate record for each session start, session end, and session reject.

2.10 Reverse Mode

Reverse mode allows you to create a single definition that identifies exceptions to a general action. You can turn on the reverse mode for one of the two session partners.

With reverse mode turned on, the defined action (PERMIT/REJECT session or alias translation) is executed for all LUs that do **not** correspond to the specified name or the specified naming mask. For example, assume that in part of a network naming conventions are not consistently in use. However, the rest of the network has naming conventions. Once you have identified the sessions within the unstructured part of the network, you can define how these sessions should be handled for the structured part of the network, and turn on the reverse mode. Thus, reducing the amount of necessary definitions normally required.

2.11 Session Management Definitions

All session management functions discussed in the previous section are determined by the definitions you provide to the *Pilot for Sessions* tables. These tables are generated by macros. To update your session management definitions, you have a choice of two methods, both of which allow the use of generic names.

2.11.1 Using the ISPF Interface

You may choose to define the table entries using the *Pilot for Sessions* ISPF interface. The interface is panel driven and the field names of each panel are self explanatory. Each session management function can easily be defined. Essentially you only have to enter the names of your LUs, VTAMs and NETs. Your definitions are used to build the assembler sources of the tables. The table definitions are completed when you choose the panel option which generates the table load modules. The ISPF interface also includes an option which allows you to test single sessions, thereby providing an immediate test facility for the new table definitions.

2.11.2 Coding the Macros

Each of the ISPF panels corresponds directly to a macro. Therefore, as an alternative to using the ISPF interface, your table definitions may also be accomplished with user coded macros.

2.11.3 Using Generic Names

Whether you choose to define table entries using the ISPF interface or by coding the macros directly, *Pilot for Sessions* allows you to use wildcard characters to define names generically. You can use generic definitions for LU names, Logmodes, COS names, VTAM names, and network names. Two wildcard characters are available:

- You can use a question mark (?) to stand for any character.
- You can use a pound sign (#) to stand for any numerical character.

Note: You cannot use generic names for adjacent SSCPs and gateway NCPs.

2.12 Dynamic Functions

Pilot for Sessions offers several dynamic functions that allow you to make immediate adjustments in *Pilot for Sessions*' operation. The dynamic functions allow you to perform the following actions:

- You can dynamically add new session definitions by generating and reloading a new control table. The new control table must have the same name as the old one.
- You can dynamically suspend *Pilot for Sessions*' operation. This function is helpful if you detect an error (for example, if *Pilot for Sessions* is obstructing the wrong sessions).
- You can issue a dynamic command to tell *Pilot for Sessions* to reject all subsequent sessions.
- *Pilot for Sessions* provides an internal session trace. The trace records are written to the system log or to a separate user defined trace file. You can dynamically turn the *Pilot for Sessions* trace on or off.

2.13 User Exits

If a session management exit already exists, it can be integrated into *Pilot for Sessions*. In addition, single session management functions can be added to *Pilot for Sessions*. This is accomplished with the use of two kinds of user exits; the global user exit and the function code user exits.

2.14 Testing Facilities

Pilot for Sessions is comprised of a number of user maintained tables in which specifically defined parameters determine session handling. All of the user definitions discussed in the preceding paragraphs of this chapter are maintained within these tables. To assist you in verifying that the definitions you have provided to these tables result in the correct handling of sessions, *Pilot for Sessions* provides you with two testing facilities:

- A production simulator for testing table definitions without establishing sessions.
- Warning mode operation for use in the production environment.

2.14.1 Simulator

The simulator function of *Pilot for Sessions* allows you to test how your table definitions effect the sessions. The simulator can either be executed in batch mode to test a large number of sessions, or it can be executed in online mode for single session checks. Input to the simulator may consist of any session.

The simulator functions just like the production version of *Pilot for Sessions* except that it does not write any records. Once you have defined all authorization parameters within a table, you may test these definitions using the simulator prior to issuing the tables to production. The simulator works with a session management exit program which coincides with the production exit. The names of two session partners are used as input. If *Pilot for Sessions* definitions are used, the result is the expected VTAM action regarding this session (PERMIT or REJECT).

Lastly, the online simulator can also be used to support the help desk in determining causes of session errors, i.e. in checking whether or not table definitions have caused session establishment errors..

2.14.2 Warning Mode

After the tables are tested with the simulator and issued to the production environment, they can again be tested using *Pilot for Sessions'* warning mode. When warning mode is active, *Pilot for Sessions* executes the necessary tests and records the results of the tests as messages. However, in warning mode, *Pilot for Sessions* does not perform the specified action. For example, if a session request generates a REJECT, *Pilot for Sessions* records the REJECT, but does not send the REJECT to VTAM. Because VTAM does not receive the REJECT, the session is allowed. When a table is being tested in warning mode, the entire table can be tested or only the individual table entries.

CHAPTER 3: COMPONENTS

The following sections describe the main components of *Pilot for Sessions*.

3.1 Load Module ISTEEXCAA

The ISTEEXCAA load module is the main program which calls all *Pilot for Sessions* functions. VTAM handles this program as its own subroutine. Therefore, you must use the name ISTEEXCAA. If you already have an active ISTEEXCAA and you wish to integrate the functions of this ISTEEXCAA with *Pilot for Sessions*, then you can define these functions to *Pilot for Sessions* as a user exit.

3.2 Load Module SMONTAB

The SMONTAB load module contains the general processing parameters used by *Pilot for Sessions*. These parameters include passwords for the dynamic functions and the name of the control table. You create the SMONTAB load module when you install *Pilot for Sessions*.

3.3 Control Table

The control table allows your site to specify how you want *Pilot for Sessions* to handle individual session requests. You create an initial control table when you install *Pilot for Sessions*. If you need to change your specifications, you can create a new control table and load the new table dynamically.

The control table is comprised of the following sub-tables:

Sub-table:	Function:
INIT	Defines standard handling for default processing of sessions.
VALIDNET	Defines the LUs that are generally permitted to have sessions throughout the network.
CHECKNOR	Defines specifically the sessions which are either permitted or rejected.
CHECKCRS	Defines the pass through (or crossing) sessions which are either permitted or rejected.
ADJSSCP	Serves as the adjacent SSCP table for defined destination or origin LUs.
GWVTAM	Serves as the adjacent SSCP table for defined destination VTAMs.
ALILU	Serves as the alias translation table for LU names.
ALICOS	Serves as the alias translation table for COS names.
ALIMODE	Serves as the alias translation table for Logmode names.
GWPATH	Serves as the gateway path list for defined LUs.

3.4 ISPF Interface

The ISPF interface consists of a program and a panel library. You can integrate it into your existing ISPF environment, if desired. Its main function is to support you in defining the *Pilot for Sessions* tables. For each panel, there is a corresponding macro that generates the specified table. The necessary macro definitions and assembly jobs are generated automatically according to the panel input. In addition, the *Pilot for Sessions* online simulator is called from the ISPF Interface.

3.5 Load Module ISTECSIM

Load module ISTECSIM is the simulator version of the main program ISTECSAA. To ensure that your table definitions result in correct session handling, the simulator may be used to test new tables. The only difference between the simulator program and the main program is that the simulator does not write any records.

CHAPTER 4: INSTALLATION AND OPERATION

The following sections summarize the tasks that are involved in installing and using *Pilot for Sessions*. Complete instructions are provided in the *Pilot for Sessions User's Guide*.

4.1 Installation

To install *Pilot for Sessions*, you copy the installation modules to the VTAMLIB. The *Pilot for Sessions* simulator and the ISPF interface may be loaded to some other user defined library. Then, to activate *Pilot for Sessions*, you restart VTAM. Or, with VTAM version 3.4.1 or higher, the activation can be done dynamically using the "MODIFY NET, EXIT" command.

4.2 Maintaining the Tables

The control table and the SMONTAB load module within *Pilot for Sessions* must be maintained by the user. As the user, you have two choices of handling table maintenance:

1. Using the ISPF Interface as a definition facility for the macros, or
2. Coding the macros directly.

4.3 Operation

Once *Pilot for Sessions* is installed and activated, your job is complete. However, if you need to modify the operation of *Pilot for Sessions*, you can use the dynamic functions described in paragraph 3.3, "Dynamic Functions", on page 8.

Only two changes require you to restart VTAM:

- Renaming the control table in SMONTAB.
- Changing the passwords for the dynamic functions.

Note: Under VTAM version 3.4.1 or higher, these changes can be activated by an exit reload by VTAM.