Configuring AutoSys and CA-7 for Cross-Platform Communication

Date Last Updated: February 18, 2004
Configuring AutoSys and CA-7 for Cross-Platform Communication

Description

In today’s IT environment there are many architectural possibilities for enterprise and non-enterprise scheduling needs. Shops with existing scheduling products may have purchased multiple engines based on business, IT, or political needs. Datacenters that are in the process of purchasing new or additional scheduling engines are doing this for essentially the same reason. This document is intended for users already running Unicenter AutoSys, Unicenter CA-7, and/or considering a purchase of either tool with the intention of running either of the products in peer-to-peer manager scenario. This scope of this document is to provide a cookbook approach on how to configure AutoSys and CA-7 to work together. It assumes that the user has already installed the products mentioned within this document in their environment and confirmed via SupportConnect or Technical Support that they are running the minimum release levels and applicable patches needed to support the configuration. The CA Unicenter scheduling engines and options that will be discussed in this document are:

- **AutoSys** – Industry leader distributed scheduling tool. The AutoSys server software can be installed on either UNIX or Windows platforms. Lightweight AutoSys agents and ERP adapters allow AutoSys to connect, schedule, and monitor to any number of different distributed and mid-range platforms along with ERP applications such as SAP, PeopleSoft or Oracle Financials.

- **CA-7** - Industry leader mainframe scheduling engine. CA-7 not only handles the mainframe environments but it also has agent capabilities that allow it to extend to distributed and mid-range platforms along with ERP applications.

- **AutoSys Connect** – Optional product that allows AutoSys to converse directly with the CA-7 instance to allow seamless dependency posting between the 2 engines. AutoSys Connect can also be used against other mainframe scheduling engines in a similar capacity.

In addition **Unicenter Enterprise Job Manager** is CA’s new Java based User Interface that allows a single ‘pane of glass’ view to manage and monitor both AutoSys and CA-7 instances. This is of tremendous benefit to scheduling administrators, management, and end-users. For administrators it allows easy of use an overall control by having a centralized view over the scheduling environment. Management and End-users can monitor mission critical workflow from the centralized view and be alerted for exception conditions. In addition, AutoSys Connect will also be discussed for those sites that currently own or are considering purchasing AutoSys Connect to work with CA-7 or possibly other 3rd party engines. This document is a living document, which implies that periodic updates may be made. This document should be used as a supplement to the other documentation for the cross platform configuration.
Architecture

Deciding how the architecture should be implemented is based on many site-specific factors. Consideration should be given to the following areas:

- Workload architecture to support the business direction
- Workload configuration to support IT structure or Line of Business requirements
- Political correctness

To ultimately determine what scheduling architecture is correct for your site please contact your local CA representative. They can assist in engaging the correct resources to help you put together a vision to ensure job management success. The Computer Associates Unicenter Enterprise Job Management solutions offer a full complement of features necessary to manage cross-system workload and dependencies.

Example 1: High-level cross platform scheduling architecture using CA-7, AutoSys, and possibly AutoSys Connect to different platforms. The AutoSys server could also run on Windows if so desired.
Example 2: A more complex high-level cross platform scheduling architecture using multiple AutoSys and CA-7 instances might look like the example below.
Configuring the Environment

In this section we will focus on installing, implementing, configuring each of the products and components to ensure success. Prior to beginning the configuration it is important to check to make sure you have the required levels, applicable patches, and a valid license key for the new platform. Current product and service pack levels are listed below. SupportConnect and/or Technical Support should be contacted to validate that there are no HYPER patches/PTFs that should be applied. For license keys there are differences between AutoSys 4.5 and previous AutoSys releases in how license keys are implemented. Contacting either the CA Licensing group or AutoSys Technical Support for help will answer any questions. For additional information in reference to specific product questions please refer to your product installation manuals or contact Technical Support. The products discussed in this document and current supported levels for this section will be:

<table>
<thead>
<tr>
<th>Product</th>
<th>Current Supported Levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>AutoSys</td>
<td>4.5, 4.0</td>
</tr>
<tr>
<td>CA-7</td>
<td>3.3</td>
</tr>
<tr>
<td>CA Common Services</td>
<td>3.0, 2.2</td>
</tr>
<tr>
<td>AutoSys Connect (optional)</td>
<td>2.4.1</td>
</tr>
</tbody>
</table>
AutoSys

In order for bi-directional scheduling to take place between AutoSys and CA-7 you must enable the Unicenter AutoSys Job Management cross platform scheduling parameter. To enable cross-platform the following Unicenter AutoSys Broker options within the AutoSys must be enabled:

- **AutoSysAgentSupport**— this activates a process managed by the Unicenter AutoSys Broker that acknowledges the mainframes presence.

- **Receive Remote Job Submission**— this activates a second process managed by the Unicenter AutoSys Broker that allows it to receive job submissions from the mainframe.

**AutoSys Instance on Windows:**

Open the Unicenter AutoSys Event Processor in the Unicenter AutoSys Administrator. To enable cross-platform scheduling support to the mainframe, check the **AutoSysAgentSupport** check box. To allow Unicenter AutoSys Job Management to receive jobs from the mainframe, check the **Receive Remote Job Submissions** check box. Both of these check boxes are located within the AutoSys Broker Options group box. See the example below:
**AutoSys Instance on UNIX**

1) To enable cross-platform scheduling support to the mainframe set the `AutoSysAgentSupport` parameter in the instance configuration file to a value of “1”.

```
$AUTOUSER/config.$AUTOSERV: AutoSysAgentSupport=1
```

2) To enable Unicenter AutoSys Job Management on Unix to receive jobs from the mainframe set the `AutoSysAgentSupportReceiveSubmit` parameter in the instance configuration file to a value of “1”.

```
$AUTOUSER/config.$AUTOSERV:AutoSysAgentSupportReceiveSubmit=1
```

**Configuring AutoSys to Monitor CA-7 Jobs**

You will need to implement support for cross-platform dependencies by creating a file named `config.EXTERNAL` (file must be defined as shown due to case sensitivity) in the `$AUTOUSER` directory. This will allow AutoSys to monitor CA-7 Jobs. Add an entry similar to the following:

```
RMT:CNCT=remote_host
```

Where **RMT** is a 3-character name that you assign to the external instance.
Where **CNCT** is set for Unicenter AutoSys Connect. (This does not mean AutoSys Connect is required).
Where **remote_host** is the **CCINAME** defined in the **SYSID** parameter of the **CCIPARM** member called by the **ENFPARMS DD** in the **TCP/IP Gateway proc** being executed on the mainframe. For further information on creating the `config.EXTERNAL` file, see the appendix "Integrating with the Mainframe and AutoSys Agents for AS/400 and OpenVMS," in the Unicenter AutoSys Job Management for Windows User Guide.

**Configuring CCI for AutoSys**

1) You need to configure CCI to communicate with the mainframe Host. On install of AutoSys, CCI creates a file named `CCIRMND.PRF` on UNIX and `CCIRMND.RC` on Windows. The file will contain a statement that defines the local machine. In order for CCI to communicate with the mainframe you will have to update this file to add a Remote statement that points to the mainframe machine.

2) Update the `CCIRMND` file in the `CCI\CAIUSER` (or `%CAILOCL0000%`) directory on NT/Windows and in the `$CAIGLBL0000/cci/config/<local host>/` folder on UNIX. Below is a description of this file.

```
LOCAL = ip-address cciname blocksize [STARTUP] [PORT=nnnn] ALIAS=xxxxxxxx
REMOTE = ip-address cciname blocksize [STARTUP] [PORT=nnnn] [RETRY=n]
```

**TCP/IP name =** Either an IP address or a name that is used as input to a name service to retrieve an IP address. You may use the TCP/IP name with the PING command to determine whether a remote connection is live. The default is the TCP/IP host name.
CCI name = A logical name CA-7 Agent may use to insulate itself from the network and any protocols being used. The default is the host name. For the REMOTE statement that defines the mainframe, the CAICCI name is the value specified by SYSID (xxxx) which by default is defined in the CCIPARM member on the //ENFPARMS DD statement in the ENF proc. The ENF joblog can also be used to determine the SYSID by checking the startup options displayed.

STARTUP = if specified, the connection is attempted as soon as CAICCI starts. This is recommended.

PORT = specifies the TCP/IP port number to use. The default is 1721, which is the default port for Unicenter TNG and CA-7 Agent. This value must match the value defined on the PROTOCOL(TCPIPGW....) statement.

ALIAS = is only valid on the LOCAL statement. If the "cciname" is longer than eight characters, a one- to eight-character alias must be used to accommodate CAICCI on the mainframe. The mainframe will truncate the name to the first 8 characters if an alias is not provided. This may cause problems if the machines have similar names.

RETRY = is only valid on the REMOTE statement. If the connection cannot be made or is broken, CAICCI will retry the connection every n minutes.

3) On a Unix machine the asbIII process communicates to the AutoSys, Universal Job Management agents, and mainframe machines using CCI. You need to configure CCI to communicate with a particular machine. For this to work you will not only have to update the ccirmtd.prf file you will also have to ensure that the cciccid.prf and the cciclnd.prf files have been created. Following is a breakdown of what these two files are for:

**cciccid.prf**
Instructs the main CAICCI demons what to do and specifies the Max_Recvrs value. The cciccid.prf file is found in the following location:

$CAIGLBL0000/ci/config/<local_host>/cciccid.prf

**Nodename** = Identifies the machine on which the Enterprise Management CAICCI demons are running. The following is an example of the cciccid.prf file:

```
CLN_Demon = cciclnd startup
RMT_Demon = ccirmtd startup
Max_Recvrs = 48,32
```

The following parameters specify what the CAICCI demons are to do and specify the Max_Recvrs parameter:

**CLN_Demon = cciclnd startup** - This setting instructs CCI to start the CCI clean demon when you start CCI.

**RMT_Demon = ccirmtd startup** - This setting instructs CCI to start the CCI remote demon when you start CCI.
Max_Recevrs = nn, mm - Value of nn defines the number of CCI receivers which also
determines the size of the shared memory segment for RVT lists. The value of mm is the
number of messages that CCI will queue up. These parameter values are explained in
the Shared Memory for RVTs section of the in the Unicenter AutoSys Job
Management for Unix Installation Guide. Note: We do not recommend that you
update the ccicccid.prf configuration file unless the file hits the Max_Recevrs limit.

cicInd.prf
Defines the number of seconds to sleep between system scans for communications
buffer and connections cleaning. The default time value for ciciInd is one second. The
default value should not be changed unless instructed by Computer Associates
Technical Support. The ciciInd.prf file is found in the following location:

$CAIGBL0000/cci/config/<nodename>/ciciInd.prf

Nodename = Identifies the machine on which the CAICCI demons are running.
All three of these files are explained in greater detail in the Unicenter AutoSys Job
Management Installation guides for Windows and UNIX.

4) CAICCI must be restarted to pick up changes to the CCIRMTD file. To Start/Stop CCI on
Windows or Unix you can use the following commands

Start CCI on Windows:
If Unicenter is installed: unicntrl start uni
If Unicenter is not installed: cciicntrl start

Stop CCI on Windows:
If Unicenter is installed: unicntrl stop uni
If Unicenter is not installed: cciicntrl stop

Start CCI on UNIX:
If Unicenter is installed: unistart cci
If Unicenter is not installed: $CAIGBL0000/cci/scripts/CCISA_rc

Stop CCI on UNIX:
If Unicenter is installed: unishutdown cci
If Unicenter is not installed: $CAIGBL0000/cci/scripts/CCISA_cshut

Defining the Mainframe Host to AutoSys without AutoSys Connect

To define the mainframe host to AutoSys database you can use jil to add the machine name
and type. From the OS command prompt enter:

1) jil
2) insert_machine: remote_host - where remote_host is the CCINAME as defined in
the CCIPARM member called by the ENFPARMS DD stmt in the TCPIP gateway proc
on the mainframe. Further explanation on the mainframe CCI/ENF parameters is
described later in this document.
3) type: t = “t” implies Unicenter AutoSys Job Management will submit work directly to
CA7.
To find out what the remote_host name is defined to in CCI, perform the following command at the command prompt:

1. `ccii` – following is a sample of the output from this task:

   Oid(XE21,CA-7 XTM UC07) Did( , ) type(R)
   Oid(XE21,SYS54) Did( , ) type(R)
   Oid(XE21,S9TR) Did( , ) type(R)
   Oid(XE21,EMSRVC_ROUTER_U) Did( , ) type(R)
   Oid(XE21,ROUTER_SERVER) Did( , ) type(R)
   Oid(XE21,S9CICICSADG3CICSADG3) Did( , ) type(R)
   Oid(XE21,JOBTRAC Server) Did( , ) type(R)
   Oid(XE21,JOBTRAC Job track) Did( , ) type(R)
   Oid(XE21,CAU9SET SetUp Mgr) Did( , ) type(R)
   Oid(XE21,SUBMITC Server) Did( , ) type(R)

   Look for the **SUBMITC Server** and just to the left of this is the **CCINAME(remote_host)** that you would use.
Adding the AutoSys Superusers, Userids, and Passwords

When an AutoSys remote agent runs a job, it logs on to the remote machine as the owner of the job. To accomplish this, the remote agent uses the encrypted passwords that were passed to it with the job request by the event processor. The event processor gets these passwords from the event server (database). Therefore, after the installation is complete and before you can run AutoSys jobs, you must enter the IDs and passwords for users who will define and run jobs. Before you can enter the user IDs and passwords in the AutoSys database, you must first establish the AutoSys edit superuser. This user is a special user to AutoSys with Administrator-like privileges. The edit superuser has read and write permissions to the AutoSys database. Only the AutoSys edit superuser can add AutoSys user passwords, using the AutoSys secure utility; however, after user IDs and passwords exist in the database, any user who knows a password can use AutoSys_secure to change that password or delete that user definition. At this time, you should also enter the AutoSys exec superuser, who can issue commands and stop the event processor.

Note: If you are upgrading, the edit superuser definition was maintained when you upgraded your database.

Defining the AutoSys edit superuser and exec superusers:

1. Open an AutoSys instance command prompt window from your AutoSys program
2. Enter the following at the command prompt - AutoSys_secure
3. When you run AutoSys_secure, a menu is displayed, and you should select the following item by entering 1: [1] Change AutoSys EDIT and EXEC superusers.
4. When prompted, enter the AutoSys edit superuser logon name and the exec superuser logon name. These users must be valid users on the machine or domain that you are logged on to.

Note: Be sure to run all AutoSys commands at an AutoSys instance command prompt, which is located in the AutoSys program group. The AutoSys instance command prompt windows set several environment variables that are needed to run AutoSys commands.
Before you can fully utilize AutoSys, you must enter the user IDs and passwords for all users on all domains that will define and run jobs. The user ID must be a valid user on the machine on which you will process the job. The user password is the same as the user’s password they use to sign on to the machine. To add a Windows user ID and password to the AutoSys database:

**Defining users associated with creating and running jobs:**

1. Log on to Windows as the user you have established as the AutoSys edit superuser.
2. Open an AutoSys instance command prompt window from your AutoSys program group. Enter the following at the command prompt: `AutoSys_secure`
3. When you run `AutoSys_secure` a menu is displayed with options to add and change users and passwords.
4. Select the following menu option by entering `4: [4] Create AutoSys User@Host or Domain password`
5. At the prompt, enter the user name, host or domain name, password, and password confirmation. If the user is created successfully, the user information is entered into the AutoSys database with the encrypted password. Windows user IDs must not exceed 20 characters, and they can include any characters except the following: (`" / ; < > | = + * … :

An example of a user would look like this: `administrator@localhost`

**Note:** Windows passwords must not exceed 14 characters, and they can contain any character except a space. In addition, they are case-sensitive and must be at least 5 characters in length. For more information on creating, changing, and deleting user IDs and passwords, see AutoSys_secure in the chapter "AutoSys Commands" in the Unicenter AutoSys Job Management for Windows and UNIX Reference Guide.
**Adding a Job to AutoSys that will Run on CA-7**

In order for Unicenter AutoSys Job Management to submit a using CA-7 the job must be defined as a valid job within CA-7. The job to be executed is specified within the AutoSys job definition. Conversely, in order for CA-7 to submit a job to AutoSys the job to be executed (specified by the SUBFILE parameter of the CA-7 job) must be defined as a valid job within the Unicenter AutoSys Job Management system. This section will help you set-up a job in AutoSys that will run under CA-7. The job type is a command job that can be defined to be dependent on certain starting conditions to instruct AutoSys when and where to run the job. To submit a job directly to CA7 where the CA7 defined job to run on the mainframe is named “CA7JOBNM” use the following jil from the OS command prompt:

1. jil
2. insert_job: anyjobname
3. command: CA7JOBNM
4. machine: cciname
5. permissions: gx,wx,mx
6. owner:user@SMFID

**Note 1:** There are 3 optional keywords that can be used on the “command” statement. They are XPSCHD, SCHID, and MONITOR. Brief descriptions of these commands are listed below but a more in-depth explanation can be found in the CA-7 section of this document under Step 2 of “Configuring CA-7 for Bi-Directional Scheduling with AutoSys”.

Example of the command statement:

```plaintext
command: CA7JOBNM ,MONITOR=monitorname, XPSCHD=xxxxxx, SCHID=XXX
```

**Monitor=monitorname**

As defined in the SVCNO parm in the CA-7 Init file. This identifies the CA-7 instance that you will pointing to and is usually only required if you are executing more then a single CA-7 instance.

**XPSCHD=xxxxxxx**

Depending on the value of the XPSCHD parameter in your CA-7 Init File you can state which method (Demand or Run) should be used when bringing a job into CA-7

**SCHID=xxxxxx**

This parameter can be used to force CA-7 to bring in a job with a specific SCHID value. If not, the default schid of 000 will be used.
CA-7

Configuration of Mainframe Common Services

Note: A recycle of CA-7 and ENF will be required to pick up changes as implemented in the next 2 sections. You do not need to recycle either product until Step 11 of 'Configuring CA-7 for Bi-Directional Scheduling with AutoSys).

1. When configuring Common Services it is a good idea beforehand to double check your release and patch levels of the mainframe Common Services (formerly CA90's) and to contact CA Technical Support for any HYPERS or recommended PTFs. Optimally your release level should be current, but try not to use anything lower than 2.2 SP1 + required patches that can be obtained via SupportConnect or Technical Support. To determine the level of CA90's you can check in the output of the CAIENF started task for message CA9075I.

2. Validate the type of TCP/IP that you are using. The type of TCP/IP used will dictate the gateway proc, CCI proc, and Common Services FMIDs to be installed. The following are the various types:
   - **TCPIPGW**: Peer to peer support for IBM TCP/IP using LE/MVS runtime.
   - **TCPIP3GW**: Peer to peer support for TCPAccess TCP/IP versions 3.1 and 4.1.
   - **TCPIPSGW**: Peer to peer support for IBM TCP/IP using SAS/C runtime in place of IBM's LE.

3. Ensure that all of required Common Services FMIDs are installed. Below is a list of these FMIDs. If any FMIDs required are currently not installed please install them before continuing. Refer to the Common Services Installation and Maintenance Guide for instructions.

<table>
<thead>
<tr>
<th>FMID</th>
<th>Common Services Product Name and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS91000</td>
<td>CAIRIM – CA Resource Initialization Manager</td>
</tr>
<tr>
<td>CW11000</td>
<td>CAIENF – CA Event Notification Facility</td>
</tr>
<tr>
<td>CW21000</td>
<td></td>
</tr>
<tr>
<td>CW41100</td>
<td>CAICCI – CA Common Communications Interface</td>
</tr>
<tr>
<td>CF33100</td>
<td>CA-C Runtime 3.1 – CA C Language Runtime Facility Release 3.1</td>
</tr>
</tbody>
</table>

**Important Note before Proceeding:**

Steps 4-5 below are only required if you are using proc CCITCPGW and the program executed CAS9CTPM, commonly known as the “C version” as it is written in the C language. The other program that may be executed is CAS9ATPM and does not require these steps.

4. Make sure IBM fix PTF UQ43223 (APAR PQ38033) has been applied and activated. It is packaged in with OS/390 release 2.10 but it is important that the TCPIP re-entrant libraries have been rebuilt. This is done by running the EDCLIB proc substituting the following exec statements for the existing //EXEC statement: (run this process once for each exec statement)
// EXEC EDCLIB,LIBRARY='TCPIP.SEZARNT1',OPARM='DIR'
// EXEC EDCLIB,LIBRARY='TCPIP.SEZARNT4',OPARM='DIR'
// EXEC EDCLIB,LIBRARY='TCPIP.SEZAROE1',OPARM='DIR'
// EXEC EDCLIB,LIBRARY='TCPIP.SEZAROE4',OPARM='DIR'

**Note:** For any release of OS/390 below 2.9., the APAR must be applied and the above proc processed substituting the above EXEC statements to rebuild the TCP/IP re-entrant libraries

5. Once the FMIDs, and if necessary the IBM APAR mentioned in #4 is linked, submit job **W010LINK** from CA Common Services sample JCL library. This job will build the CCI Gateway load module required for cross-platform communication.

6. Update the proper TCP/IP proc to be used as the gateway to site standards. The procedures containing the JCL needed to set-up the TCP/IP gateway are normally found in the CA Common Services CAIPROC library as CCITCPGW.

7. Update the ENFPARMS DD statement to point to the CCIPARM member that will execute the proper TCP/IP gateway proc. The information contained in the CCIPARM member will include a SYSID and a Protocol statement as defined below:

```
SYSID(cciname)
cciname is the name that CAICCI will use for this node. This value is a unique name and usually is the same as the LPAR. It can be any unique name up to 8 characters long.

PROTOCOL(TCPIPGW)
TCPIPGW – Peer-to-peer support for IBM TCP/IP using LE/MVS runtime.
Proc: CCITCPGW.

**Note:** Other parameters, such as the port address may be required to be added to the PROTOCOL statement. This would depend on whether the site used installation defaults or modified proc names.

8. Update the proc to conform to site standards and copy to a system proclib.

9. Start or Restart CAIENF to have changes take effect. If ENF cannot be recycled a modify command "F ENF,PROTOCOL(TCPIPGW,7000)" may be issued to start the gateway proc. The 7000 is the port number for the proc to use and can be changed to site standards.

10. **Note:** After restarting ENF, the TCPIP gateway proc will execute. If this proc abends with a SOC4 verify that the above-mentioned IBM PTF(UQ43223) has been applied and the TCPIP re-entrant libraries rebuilt as mentioned.
Configuring CA-7 for Bi-Directional Scheduling with AutoSys

1. Define the CAIENF Cross-Platform Scheduling Feedback Event (CAXPSFBK). You will need to update and run CA-7 SAMPJCL member L2DCM2. This job will define the CAXPSFBK and CAIENF event and data elements. This will require a recycle of ENF after the ENF event is defined (See Step 11).

2. Validate the correct cross-platform scheduling keywords are defined to the SVCNO statement in the CA-7 Initialization File. An example of the SVCNO statement is below

```
SVCNO, SASSVC=YES, ROUTER=YES, MONITOR=YES, XPSSCHD=RUNREF
```

**MONITOR=YES**: This indicates that CA-7 XPS SERVER functions should be activated. **MONITOR=YES** will use CA7xxxx (where xxxx is the smfid of the local system) as the Monitor Name. If you would like you can explicitly define the monitor name (MONITOR=CA7xxxx) in place of MONITOR=YES. Where xxxx is the smfid or four unique characters. If you are running multiple CA-7’s you will need to explicitly identify the CA-7 being used. The XPS ROUTER will execute on only one copy of CA-7 at any given CAICCI node. The XPS ROUTER will be started on the production copy of CA-7 by default if **MONITOR=YES** is coded. If cross-platform scheduling is to be used only with a test copy of CA-7 then **ROUTER=YES** must be coded on the SVCNO statement for the test copy in order to start the XPS ROUTER.

If the SVCNO parameter in the CA-7 initialization file (ONLINE) is set to **XPSSCHD=RUNREF** you must add the XPSSCHD parameter to the end of the command info in AutoSys. This will allow the job being submitted by AutoSys to act as if it was a normal demand. If the XPSSCHD keyword on the SVCNO parameter is set to **DEMAND** the above keyword is not necessary. By adding **SCHID=XXX** (where XXX = a specific CA-7 schid) to the AutoSys command you can request a specific SCHID to be used when bringing the CA-7 job in. If no **SCHID** is used CA-7 will bring the job in under **SCHID=001** (default SCHID). The XPSSCHD parameter defines how the job behaves under CA-7. By default, cross-platform jobs enter CA-7 using the **RUNREF** option. This assigns the role of scheduling manager to AutoSys. The primary responsibility for workload control belongs to AutoSys. Jobs scheduled using this option will NOT honor requirements defined in CA-7, they will NOT be considered requirements for other CA-7 jobs and they will NOT trigger other CA-7 jobs at completion. This variant of the RUN command differs from the standard RUN command in that it will not allow a CA-7 restart. A job scheduled using this command is considered "complete" at either normal or abnormal termination. An entry for the job will appear in the CA-7 RUNLOG, however no entry is created for the job in the prior-run queue.

A greater degree of workload control over cross-platform jobs may be assigned to CA-7 using the **XPSSCHD=DEMAND** or **XPSSCHD=RUN** options. If either of these options is used, then additional management functions become the responsibility of CA-7. **XPSSCHD=RUN** confers additional responsibility on CA-7 for monitoring and control of restart and rerun conditions. However, since the RUN command is used to schedule the job, CA-7 requirement and trigger definitions will be ignored. **XPSSCHD=DEMAND** confers even more management responsibility on CA-7. Because the DEMAND command is used to schedule the job, CA-7 requirement and trigger definitions will be honored, and
CA-7 must be used to monitor restart and rerun conditions. It is recommended that `XPSSCHD=DEMAND` be used to provide the extra control.

3. You will need to define at least one ARF trailer terminal. This is a `DEVICE=TRXDV` type terminal. If you all ready have one defined, you do not have to define another. Here is a sample definition, which consist of 4 statements:

```
GROUP,NAME=TRXGRP,DEVICE=TRXDV,LNAME=TRXLN,OPEN=YES
LINE,NAME=TRXLN,BUFSIZE=1024,TNAME=TRXT1,OPEN=YES
TERM,NAME=TRXT1,DEVICE=TRXDV,NLINE=1,LINLEN=80
STATIONS,TRMID=TRXT1,STANIDS=(TRK)
```

**Note 1:** See *CA-7 Systems Programmer Guide* for additional information on defining terminals.

**Note 2:** CA-7 will need to be recycled for the options added in steps 3 and 4 (see Step 12) to take affect. When CA-7 starts up you will see `CAXPSxxx` messages in the joblog.

4. If you wish to define cross-platform scheduling password requirement rules then you must create a data set or PDS member to hold these rules. Add an `XPSPSWD DD` statement to the CA-7 JCL where the XPS ROUTER will execute. Review "Cross-Platform Server Password Requirements" in Section 6.3.3 in the *Unicenter CA-7 Interfaces Guide* on coding the password requirement rules. This is optional and if you make this change then a recycle of CA-7 will be required (see Step 11).

5. Allocate and initialize the CA-7 XPS Profile PDS. You can complete this by modifying and executing member `XPSPROF` from the CA-7 sample JCL library.

6. Create member `CACCENV` in the XPS Profile PDS with the keyword entry specifying the environment parameter for the submit function. This variable is the Monitor Statement, which must be exactly seven characters. `CA7PROD` should be avoided because Unicenter TNG and CA-7 commonly use this for VAX. A good choice might be `CA7` followed by the SMF-ID of the originating system. This value must match the one used by the CA-7 Cross-Platform Tracking System (`CA7XTRK`) as its parm value.

7. Allocate the CA-7 XTRK Checkpoint files to be used in the CA-7 cross-platform tracking STC. Update member `XPSCKPT` in the CA-7 Sample JCL library. See "CA-7 Cross-Platform Tracking JCL" in section 6.2.2.1 of the *CA-7 Interfaces Guide* for more information.

8. Modify member `CA7XTRK`, in the CA-7 sample JCL library, with the seven-character parm value defined in the MONITOR statement in member `CACCENV` that was created in Step 6 above. Once this has been completed, move the started task JCL to the site standard proclib and start CA-7 cross-platform tracking task (`CA7XTRK`). `CA7XTRK` can be also run as a subtask under ICOM. See "CA-7 Cross-Platform Tracking JCL" in section 6.2.2.1 of the *CA-7 Interfaces Guide* for more information.

9. Modify member `CA7TOUNI` in the CA-7 sample JCL library to define the CA-Driver procedure for the CA-7 XPS submit function. This proc will enable submission of a CA-7 batch job that will execute program `CA7TOUNI` to send execution data through CAICCI to
the XPS SERVER on the target platform. The XPS SERVER on that platform in turn submits the job and returns CAIENF tracking data as it executes. Any output from the job will be directed to the XPS SERVER console on that platform. The PROFILE DD must reference the XPS Profile Dataset defined in step 5.

10. Allocate a CA-7 Driver proc and add a CARPROC DD statement to the CA-7 Online task. Copy member CA7TOUNI into this driver proc. CA-7 must be restarted for changes to take effect (see Step 11). If you are not familiar with CA-Driver, see "CA-Driver Procedures, Variable Parameters, and Functions" in Section 4.0 of the CA-7 Interfaces Guide.

11. Recycle CA-7 at this time to allow for all CA-7 changes to take effect

12. Define the CA-7 cross-platform jobs and JCL for the submit function. You can use member CA7XSUB as a guide. The jobs should be defined and scheduled like any other CA-7 job, only the JCL is different. There are multiple parameters that can be defined within the SYSIN DD of the cross platform job. The two required parms are the NODE and the SUBFILE parameters that identify the destination where the job will be submitted and the job name that will be run in AUTOSYS. There maybe other optional parameters that might be applicable to your environment. Review the Submit Function section of the CA-7 Interfaces Guide. An example of the JCL is as follows:

```
#7UNI
  //CA7TEST1 JOB ...jobcard...values...
  //STEP1   EXEC  CA7TOUNI    (execute CA-Driver proc)
  //SYSIN    DD *
NODe=cctiname
SUBFILE=jobname
/*
Note 1: The jobname is case sensitive. For more information see "CA-7 Cross-Platform Tracking JCL" in Section 6.2.2.1 of the CA-7 Interfaces Guide for more information.
AutoSys Connect (optional)

The benefit of having AutoSys Connect is that job/event dependencies associated with CA-7 submitted jobs can be automatically tracked and real-time posted without requiring an additional AutoSys job to be defined and executed to initiate a CA-7 job.

AutoSys Connect Features

XCCODE

One of the benefits of using AutoSys Connect is to allow for the return of true job status. For example, a job running on a distributed machine is believed to be a failed job unless it returns a code of 0. With AutoSys we can customize the job stream through exit codes of certain jobs and allow the job stream to continue processing. Consider the following scenario:

You want AutoSys as_jobB to process on the success condition of ca7_jobA however, CA7JOBA does not simply return 0 as a return code. Within the JCL of this job there are multiple job steps, and we can define through the XCCODE member on which job steps we would like Connect to return failure back to AutoSys.

c7_jobA ends on step PAYROLL with a return code of 4. According to this particular c7 job, the job finishes successfully but with a warning however, on the distributed side, this return code will still indicate a failure. To define specifically what exit codes you would prefer to return failure, you will need to modify the XCCODE member located the AutoSys Connect parmlib on the mainframe. XCCODE entries will be defined in this member and will have the following format:

```
JOB <jobname> STEP <stepname> RC <operator>
<return_code>
```

Now you want the job to return success if the return code for STEP PAYROLL is less than 4. So we would define our condition line in the XCCODE member to tell Connect when this job will fail.
We want the job to fail when the payroll step is greater than or equal to 4. We could input the following line in the member.

<table>
<thead>
<tr>
<th>JOB</th>
<th>CA7JOBA</th>
<th>STEP</th>
<th>PAYROLL</th>
<th>RC</th>
<th>GE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For more information XCCODE usage, please refer to the AutoSys Job Management Connect Option under the section titled Operations.

**AutoSys Connect Post-Installation Steps**

1. After installing AutoSys Connect start the AutoSys Connect STC and verify that it is running through CCI on the distributed machine. By default the STC name is “CAUATCON” (CCI can be installed and configured by referencing the previous section “CONFIGURING CCI FOR AUTOSYS”). The output should list AutoSys Connect as one of the tasks running on the mainframe. An example is below:

   Oid(A71SENF,CPS Manager ) Did( , ) type(R)
   Oid(A71SENF,AUTOSYS-CONNECT ) Did( , ) type(R)
   Oid(A71SENF,W410_SPAWN_SERVER ) Did( , ) type(R)
   Oid(A71SENF,MVS_START_SERVER ) Did( , ) type(R)
   Oid(A71SENF,CA7XE71  Job track ) Did( , ) type(R)

   **Note 1:** If the mainframe machine does not appear then double-check your configuration of CAICCI. If the machine does appear but AutoSys Connect receiver does not then check the CAUATCON STC output for additional error messages.

   **Note 2:** AutoSys 4.0 and 4.5 require CCI to be active in order to work correctly. It is suggested to add the start command to the ENF commands member to ensure that CAICCI has initialized completely.

**Configuring AutoSys to Monitor CA-7 Jobs**

This step will demonstrate how to create an AutoSys job that will depend upon the condition of a CA-7 job. For example, you will need to define an AutoSys job that will depend on the condition of a mainframe job that runs through CA7.

The condition field for the AutoSys job will be defined using jil similar to the one below:

```
insert_job: cnct_dep
job_type: c
machine: localhost
condition: success(MFJOBNAME^CA7)
```

When this job gets submitted to the database successfully, AutoSys will send an external dependency request to Connect telling it that it wants to receive the all JOBINIT and JOBTERM messages for that particular job. You should see the following output in the AutoSyslog output.

```
[10:22:01.2100] [1] ----------------------< Date: 01/13/2004  10:22:00 >----------------------
```
When CA-7 submits the job, AutoSys will see the running events and display them in the AutoSyslog as shown below:

When CA-7 submits the job, AutoSys will see the running events and display them in the AutoSyslog as shown below:

```
[10:32:01.1900] [1] ----------------------< Date: 01/13/2004 10:32:00 >-----------------------
[10:32:32.5070] [1] [localhost connected]
[10:33:01.1900] [1] ----------------------< Date: 01/13/2004 10:33:00 >-----------------------
```

Configuring AutoSys to Submit Jobs Utilizing Connect

In addition to being able to monitor jobs on the mainframe, you can initiate jobs from AutoSys to the mainframe through Connect.

The auto_cnct command is used to define command jobs that will be run through Connect. The syntax of the command is as follows:

```
auto_cnct -a <node> -s <scheduler code> -j <jobname> -c <command> -p 'parameters' -d
```

**Note:** A description of these parameters and examples are available in the *Unicenter AutoSys Job Management Connect Option Installation and Administration Guide* under Appendix C.

You will need to define an AutoSys job using jil and specify this command in the command attribute of the AutoSys job. An example is below:

```
insert_job: cnct_job
job_type: c
command: auto_cnct -a A71SENF -s CA7 -j PINVM001 -c RUN -p 'SCHID=1' -d
machine: <cciname>
```

**Note:** You do not need to specify the owner of the job in the job attribute to be a user on the mainframe machine. If the user has the credentials to run the job, once the job is submitted via the AutoSys broker, Connect relies on the logon permissions executed by the Connect’s batch interface JCL located in the Connect PARMLIB on the mainframe.
High Availability Support

Unicenter AutoSys Connect 2.4.1 also provides support for a high availability support for AutoSys to ensure that messages reach their intended destination. This is used during a time of failover for when Connect can no longer communicate with the primary EP machine, it will automatically route queued messages to the shadow machine after the shadow EP has taken over in a failover situation. This process is documented in detail in the AutoSys Job Management Connect Option under the section titled Operations.