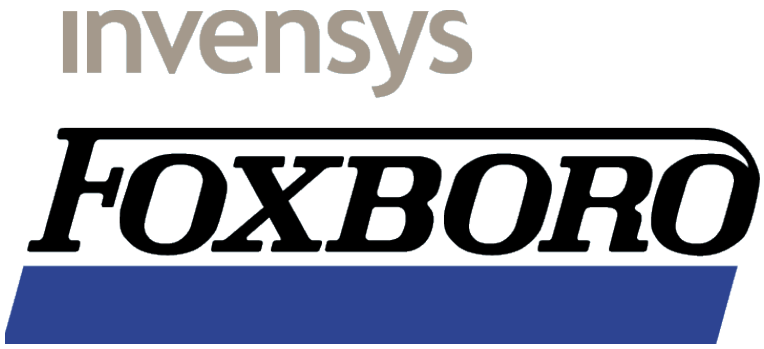


OSI PI on I/A 50 Series systems

Installing and configuring the OSI PI interface software

Ron Deen / Ron.Deen@IPS.Invensys.com

Revision 2.3
January 30, 2006

Contents

1 Introduction

The PI interface to the Foxboro I/A Series systems provides for the bi-directional transfer of data between Foxboro I/A Series computers (through FoxAPI) systems. This document attempts to be a guide on how to get the OSI PI software installed and running on the I/A Series platform. This document covers the installation, user account setup and network configuration required for this task.

Please note that this is NOT an Official Invensys or Foxboro manual.

Configuring the PI software itself is NOT covered in this document. I.e.: how to get the tags from the Foxboro system over to the PI system is assumed to be within the scope of a third party.

If you have any information that would make this document suite your needs better, please feel free to let me know. I can be reached at the e-mail address on the cover.

1.1 Disclaimer

Due to the inherently complex nature of computer software, Foxboro does not warrant that the software described in this document or this documentation is completely error free, will operate without interruption, is compatible with all equipment and software configurations, or will otherwise meet your needs. Accordingly, this documentation is provided as-is, and you assume all risks associated with it's use. Invensys makes no warranties expressed or implied, with respect to this document. In no event will Invensys be liable for indirect, incidental or consequential damages, including, without limitation, loss of income, use, or information.

1.2 Prerequisites.

You are familiar with the Foxboro standard system software and hardware and are able to setup and configure the additional hardware that may be required to implement the PI software. Some scripting capabilities may come in handy as does familiarity with the Foxboro Integrated Control Configurator. The PI software described here is assumed to be installed on a AW/AP 51 style A or higher processor.

The OSI PI software consists of two basic parts:

- The Interface between the the Pi client and the PI Server. This part takes care of the communication between the PI Client and the remote PI Server application.
- The Interface between the PI API and the FoxAPI server referred to as "`piapi`".

Apart from this, this document assumes the following:

- Both these interfaces are assumed to be installed in the same directory structure.
- In the setup we use `/opt/piapi` as the base directory for the PI software.
- The FoxAPI program must be installed and configured properly.
- You have the OSI PI installation software and documentation at hand.

1.3 What you should have.

In order to be able to install and configure everything according the procedure, you should have the following:

- An I/A Series system with software version 4.3 or higher, configured with a secondary Ethernet connection.
- The OSI PI Software suitable for the platform you are installing the client on (Solaris, NT, XP).
- A configured/installed current version of FoxAPI.
  At the time of writing, FoxAPI 4.2.8 is assumed to be used. It is available through your Invensys representative.
- Access to a CD-Rom drive and a floppy drive.
- The (customer supplied) information regarding IP addresses, Routers, *netmasks* etc. for the second Ethernet connection. Also the name and IP address for the PI Server since this is required on the Client side.
- Approximately 30Mb space on the system disk for the PI programs + additional space for the buffering capability of PI.
- The installation software from OSI that includes the `fxbais` interface and the PI interface.

1.4 Conventions

When you see this:	***It means this:***
[Filename]	This typeface indicates a filename of which the contents start on the next line. This line is not part of the file contents
`STATION# command to type<cr>`	This typeface is text as shown on screen or when placed under [Filename] indicated the file contents
Follow these steps	Indicates a procedure to follow
`<Alt_F4>`	Text between <> signs indicate a key combination: Press the Alt key together with the Function key F4 in this case

1.5 Revision history

Revision number:	***Description***
Revision 1.0	Initial release
Revision 1.1	Some additions, Fixes, typos etc. Added more info for user setup. Added command line useradd.
Revision 1.2	Major work on content done. Added FoxAPI configuration info.
Revision 1.3	Typo's fixed etc.
Revision 1.4	Added check for FoxAPI in go_pistart script from Foxboro
Revision 1.5	Minor cleanup, (Typo's etc)
Revision 2.0	Invensys standard layout. OpenOffice.org for Maintenance. Total work over.
Revision 2.0.1	Minor changes. Section 5.6 is now titled Installing the PI AIS interface (`fxbais`). Using the PISERVER name consistantly.

Revision number:	***Description***
Revision 2.1	Some small changes to the `go_pistart` script.
Revision 2.2	Found out that `user_apps.dat` is not always in `/etc/fox`. So noted. Changed the title. It is for 50 Series only so changed that…
Revision 2.3	Integrated some feedback: Rotating PIMESSLOG file (5.7.4 Rotating the PIMESSLOG file.) according PI manual, (Thanks Francois Le Garsmeur).

2 Why do we have this document?

Basically we (OK, it's just me) have found some things done through the OSI way, that are not done in the way we (I again) would like them to be done. This chapter goes into this, and hopefully provides an alternative that avoids conflicts with what I think is the proper approach.

2.1 What is the problem with doing things the "OSI way"?

Please note this is the author's opinion only and not necessarily the Foxboro or Invensys view. I do believe that the OSI way of implementing PI does conflict with some ways we expect things to go. This may lead to confusion and may even lead to a "time bomb" type of implementation where PI will run properly for a long time and suddenly after installing a totally unrelated piece of software, things are broken. Now we don't want that, do we?

Here is what I disagree on with OSI:

- **First**:
  OSI suggests you should create and use a *non-root* user account. Although I share their thoughts in this area but there is a *catch*. The FoxAPI program will not talk to anyone BUT root. FoxAPI is like that by design.
  So you end up with setting the UID of the suggested "*piadmin*" user to 0, which makes it the same as root, ruling out a lot of the benefits of having a non-root account along the way.
- **Second:**
  The standard OSI *pi-start* script will try to deterime if FoxAPI is running which is fine. When it is not running however, the script will try to start the FoxAPI interface. I do not consider this proper behavior. FoxAPI is important the I/A Series system but to my believe it is not third party business to start or stop any API on the system.
- **Third:**
  OSI suggest to switch away from the Foxboro way of starting third party applications on our system. Again I do not agree with that. We have a well documented procedure how to start third party applications installed on our platform and these are expected to be installed accordingly. This avoids confusion for anyone working on the I/A Series system.
- **Fourth:**
  The standard OSI script will check for a process with the name "`om_poll`" and when this is present, decides that all systems are go, because FoxAPI is running apparently.
  This can go wrong in two ways:
  The test will pass even if AIMAPI is running because this too has an "`om_poll`" process.
  FoxAPI is however NOT running so PI has no way to get data out of the system.
  BUT WHAT IS EVEN WORSE MAYBE:
  When BOTH FoxAPI and AIMAPI are configured AND running correctly, the PI software will fail because more than one instance of "`om_poll`" is found. You can have both AIMAPI and FoxAPI on a system and this is very normal behavior for an I/A Series system.

As a result I have put together this document as a guide to get the OSI PI software properly installed on I/A Series systems and do this without sacrificing PI functionality.

Installing PI and making it work involves a little more that just installing PI. Some other parts of the I/A Series system need to be setup in order to make things work. This includes setting up FoxAPI and your network settings for that matter. This document will attempt to address all these issues.

The steps to be taken to accomplish a successful installation are:

- Setup the network configuration.
- Adjust the installation (if required) to install FoxAPI and configure the package.
- Optionally create the account for the "**piadmin**" user and allow "root" privileges to this account.
- Install the OSI software (both **fxbais** and **piapi**) using the "root" account (as described in the OSI documentation).
- Configure the system to automatically start the PI software at boot time.

At least the files indicated here are subject to modification:

Network settings files:

- `/etc/hosts` where the hosts (for instance the PI Server) are defined.
- `/etc/hostname.xxx` The second Ethernet name of your AW or AP
- `/etc/defaultrouter` If routing is in place, what is the default route.
- `/etc/netmasks` which host IP addresses are within your network address.

Note: `/etc/netmasks` on Solaris 2.5.1 currently only supports class A ,B and C networks (no other subnets) in the currently used Solaris version (2.5.1). This implies that setting an **netmask** like 255.255.255.248, will not work, the OS will default to the default **netmask** that "belongs" to the IP address specified.

3 Installing and configuring FoxAPI

The FoxAPI package is installed through the System Definition install process. Consult your Foxboro representative if you need the package installed. The FoxAPI interface requires the user to have “root” privileges. This would require you to either be 'root” when exchanging data through FoxAPI or give the user these rights in the `/etc/passwd` file. The FoxAPI programs are located in `/opt/fox/ais/bin` directory.

3.1 FoxAPI installation

Follow the instructions that come with FoxAPI if it not already installed on your system. At the time of writing the latest version of FoxAPI was released as a QuickFix. Follow the instructions in the QF to get FoxAPI installed on your system. When you want to add FoxAPI installed properly (and Yes, that IS what you want) you should add the package **ADDE6** for Solaris or **ADDE7** for the Windows NT or Windows XP platform on the host platform. Depending on your hardware platform there is a specific QF number that applies:

- Solaris 2.5.1 QF1005346
- Solaris 8 QF1005387
- Windows NT/XP QF1005388

At time of writing these were the current ones. These latest FoxAPI quickfixes are available through your Foxboro representative.

3.2 FoxAPI configuration changes for PI

There are some small modifications required when you want PI and FoxAPI to work together. This is done the FoxAPI configuration file. In the directory `/opt/fox/ais/bin` you will find a file with the name “`foxapi.cfg`”. If it does not exist you must create it. This file should contain some mandatory settings for FoxAPI to work with PI.

Special instructions for PI applications:
Three new features were added in FoxAPI 4.2.6. These features are activated by default. However these prevent PI applications from functioning and thus they should be deactivated by editing `foxapi.cfg` to include the following lines:

[/opt/fox/ais/bin/foxapi.cfg]
```
ia_badstat=0
skip_omread=0
protect_index=0
```

The required settings can be retrieved from the accompanying documentation.

4 Setting up the network.

To have access to both the PI server and the Foxboro 50 Series machine, the network needs to be configured. We use the optional second Ethernet interface to communicate with external sources. Only the Foxboro side is covered here. On the I/A 50 Series an additional Ethernet interface needs to be installed. Please consult your Foxboro representative for the possible alternatives. When the network hardware is properly installed we have to define/add the name of the Foxboro server and add that name to the `/etc/hosts` table and we must create a file that links this name to a network interface. The file could be something like this assuming the 2nd Ethernet name would be **JIMBOB**:

[/etc/hosts]

```
#
# Internet host table
#
127.0.0.1      localhost      loghost hw1197
#*****************************************************
# SECOND Ethernet hosts
#*****************************************************
10.10.10.1     ROUTER
10.10.10.10    JIMBOB
168.120.168.45 PISERVER
#*****************************************************
# SECOND Ethernet hosts
#*****************************************************
#*****************************************************
# Start of I/A hosts
# created Fri Mar 2 14:54:20 GMT 2001
#*****************************************************
#
# The following host entries were created by the I/A
# Software Install sub-system. Any additional entries
# should be placed AFTER the End delimiter.
#
151.128.16.66  1AP501
151.128.16.68  1AW701
151.128.16.67  1AW702
151.128.16.65  1AWB11  loghost
151.128.16.134 2WP7E1
151.128.16.132 2WPAE1
151.128.16.131 2WPDE1
#
#*****************************************************
# End of I/A hosts
#*****************************************************
```

The file `/etc/hostname.le1` contains the name for this host (as mentioned in `/etc/hosts`) on the secondary Ethernet connection and relates this name to the network hardware though the file name. So this file contains the name JIMBOB

[/etc/hostname.le1]

```
JIMBOB
```

Do NOT alter the file `/etc/hostname.le0` because the adapter associated with this file is used for the Foxboro I/A nodebus exclusively.

In this example, a router (named ROUTER) was added. This router is required since the PISERVER is not within the *netmask* of the secondary Ethernet.

To make a router work and make the system use this route to find it's way outside the network, we must make this router the "Default router".

These are the steps to accomplish this:

- Make sure that the IP address and name for the router are inserted in the host table. In this example the name ROUTER is used with IP 10.10.10.1.
- If it does not exist, create a file in `/etc` named: `defaultrouter` containing the name of the router you are intending to use as the default router.
  To create this file type, logged in as root, type at the prompt:

  ```
  # echo ROUTER > /etc/defaultrouter<cr>
  ```

  At boot time the system will search for this file and when it finds this file, will setup the default route for you.
  To check if this has worked without booting your Foxboro box try this at the prompt:
  ```
  # route -f add default ROUTER 1
  ```
  If all is well you should be able to ping the ROUTER and if the ROUTER is properly configured you must be able to successfully ping the PISERVER from JIMBOB as well.
- If the PISERVER address is within the default *netmask* you may need to narrow down the *netmask* to force the route when accessing the PISERVER.

When you have reached this part of the document, but haven't got a clue how you got here, I really must suggest you get someone to help you out.

5 Installing the PI software.

Installation of the OSI PI software must be done with the *root* privileges so: Login as "**root**".
Installing PI software basically implies: copy the files to a temp directory (`/opt/tmp` for example) and uncompress the files there. I have seen some problems with this:

- The compressed files have an extension with a lower case "z". This is not correct and this must be altered to an upper case "Z".
- The command provided by OSI does not work safe.
  The command as found in the OSI documentation: "`zcat /cdrom/fxbais_tar.Z | tar -xvf -`" produces some undesired file corruptions. Better take the two step approach and run the two commands separately (uncompress first and after that, extract the tar file)

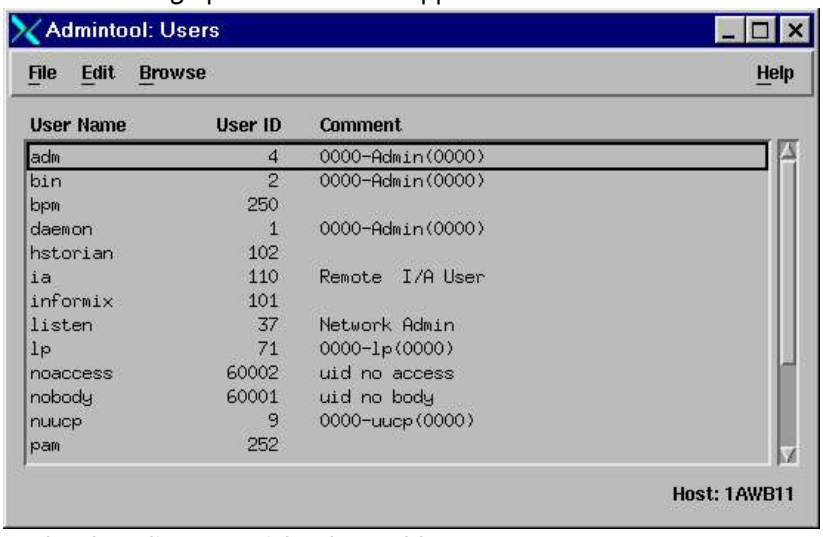
In this section we take the cautious road ahead.

5.1 Optionally create the "piadmin" account.

If you decide to run PI under another user account than "root", you could follow these steps before you install the PI software Here we assume you will choose the account named "piadmin" which seems a logical name considering the function of this account. Note that this account will be used to run PI, you always must install PI as "root".
This user account is of course not a standard Foxboro account and thus must be created. In this example I chose to use the SUN X application "*admintool*" for this purpose. You do whatever you want to add the user account, as long as you feel comfortable with the procedure you choose.
Remember that the user MUST have "root" privileges to interact with FoxAPI.
In a VT100 screen on the AW execute the command "*admintool*"

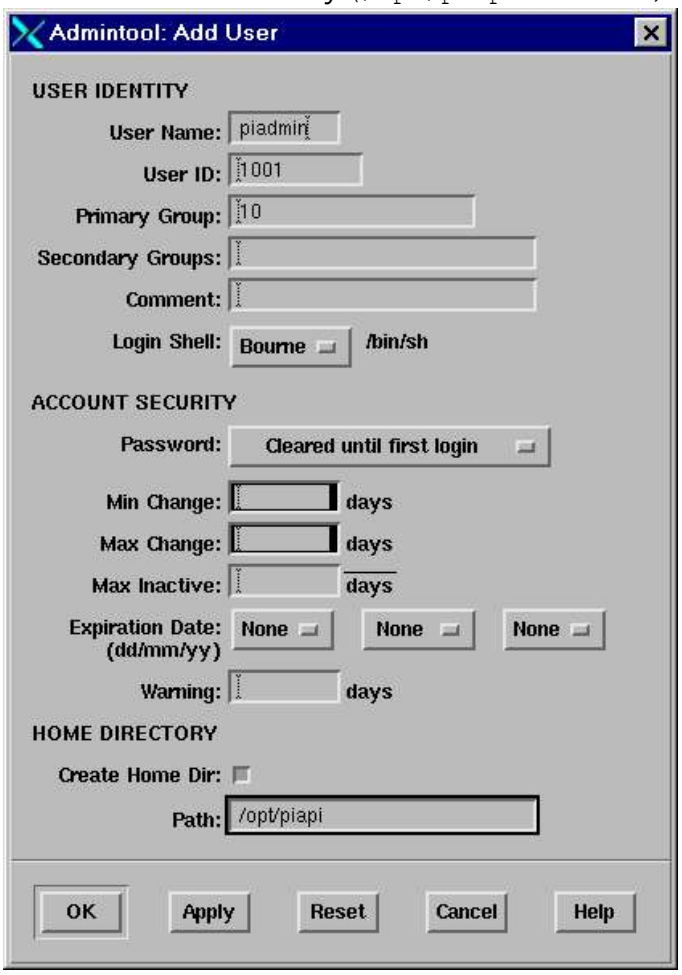
```
# admintool &<cr>
```

which will bring up the **admintool** application.



Under the Edit menu pick select Add... to create a new user account. Enter the fields as indicated

and select an home directory (`/opt/piapi` in this case)



Adding the account in this way, will result in some warnings as the primary group is "**staff**"
The default shell is the *bourne* shell which is nice because most shell scripts used by both Foxboro and OSI are *bourne* shell scripts. After this it is important that you manually set the UID of the *piadmin* user to 0 to give this user "root" privileges.
The line in the `/etc/passwd` file could resemble something like this:

```
piadmin:x:0:1::/opt/piapi:/sbin/sh
```

5.2 The OSI PI shell environments.

The **PI** software requires some shell variables to be set at login time. These variables include the `$PIHOME` and the `$LD_LIBRARY_PATH` variable. Further it may be nice to have the MAN pages at hand, so the `$MANPATH` variable must be set also.
The **PI** software has the `/opt/piapi` directory for home and uses the *Bourne* shell by default. As a consequence of this *Bourne* shell, a `.profile` file defines the environment. This `.profile` file is created in the **$PIHOME** directory.
The contents of this file are:

```
[/opt/piapi/.profile]
PATH=/bin:/usr/bin:/usr/sbin:/etc:/usr/ucb:/usr/local:/usr/ccs/bin:.:/usr/opt/SUNW
md/sbin:/opt/piapi ; export PATH
PIHOME=/opt/piapi ; export PIHOME
LD_LIBRARY_PATH=$PIHOME/lib:$LD_LIBRARY_PATH ; export LD_LIBRARY_PATH
MANPATH=/opt/share/man;export MANPATH
```

If the "**root**" account is also accessing the **PI** software, the variables must also be set for this account. For "**root**" these are set in the "`/.cshrc`" file since "**root**" uses the "**csh**" shell by default.

```
[/.cshrc]
set path=(/bin /usr/bin /usr/sbin /etc /usr/ucb /usr/local /usr/ccs/bin . /
usr/opt/SUNWmd/sbin /usr/foxbin /opt/fox/bin/tools /usr/fox/wp/bin/tools /
opt/piapi)
setenv PIHOME $PATH;/opt/piapi
setenv LD_LIBRARY_PATH $PIHOME/lib
setenv MANPATH /opt/share/man
```

These changes will allow both **piadmin** and **root** to access the **PI** programs.

5.3 Installing PI API.

To install the PI API software we must copy the compressed tar file to a location on the system drive. This tar file looks something like `piapi134r1_sol2_tar.z` . Again note the lower case "z" in the compressed tarball, which needs to be made upper case ("Z").
First copy the tar file to a temporary locations like `/opt/tmp` and uncompress it:

```
# cd /opt/tmp<cr>
# cp /cdrom/piapi134rl_sol2_tar.Z .<cr>
# uncompress piapi134rl_sol2_tar.Z <cr>
```

After completing this go to the directory where you plan to install the PI software. In this example we chose the `/opt/piapi` directory. Go to that directory and extract the tar file from this location:

```
# cd /opt/piapi<cr>
# tar xvf /opt/tmp/piapi134rl_sol2_tar <cr>
```

After this, your `/opt/piapi` directory should have these contents:

```
1AWB01# pwd<cr>
/opt/piapi
1AWB01# ls -l<cr>
total 6
-r--r--r--   1 545      545         1960 Sep  6  2000 PIAPIunix.txt
drwxr-xr-x   3 545      545         1024 Oct 28  2000 build
1AWB01#
```

When this is done, we should have in the directory `/opt/piapi/build` an install script that will install the PI Client interface on this system. The script to install PI is **`pi.install`** and this must be executed next:u

5.4 Setting up the PI Client software

To setup the PI Client software we must run the install script located in `/opt/piapi/build`, the install script is named `pi.install` :

```
# cd build<cr>
1AWB01# ls<cr>
api          rel110.txt   rel121.txt   rel1233.txt  rel12b3.txt  rel132.txt
pi.install   rel112.txt   rel122.txt   rel1234.txt  rel130.txt   rel133.txt
rel104.txt   rel113.txt   rel123.txt   rel1235.txt  rel131.txt   rel134.txt
rel105.txt   rel114.txt   rel1231.txt  rel1236.txt  rel1312.txt
rel106.txt   rel120.txt   rel1232.txt  rel12b2.txt  rel1313.txt
1AWB01#
```

The `pi.install` script is the one we need to execute. Make sure to do this as "root" and make sure to do this in the "Bourne shell". You will get an error stating: **Variable syntax** when using the "C-shell"

5.5 Executing pi.install

As mentioned, `pi.install` is the script we must run as "**root**" to install the first part of the OSI software. This will bring up the dialog to configure the PIAPI settings. Marked in **blue** you will find the user responses:

```
# pi.install<cr>
The PIHOME environment variable has not been defined
Please enter the path for the PI-API installation:
/opt/piapi<cr>
Creating/Updating the PI-API file system
PIHOME is properly defined: /opt/piapi
Setting PI Environment Variables
Installing PI-API from /opt/piapi/build
BLDDIR exists
LIBDIR created
BINDIR created
DATDIR created
Enter an existing user name for PI-API [piadmin] ?
piadmin<cr>

User name: piadmin
Creating File: piclient.ini
Please Enter the Node Name of the Default PI Home Node:
PISERVER
Is PISERVER a PI3 (UNIX, NT) system? [Y,N]
Y<cr>
Home node: PISERVER
PI3 node: y
Installing iorates.dat

SUN has changed the binary output of objects with Compiler SC5.
If you have programs compiled with version 4 or earlier compilers,
you should install the CC4 version of the PI-API library.
If your program documentation does not mention SC5 compatibility,
you should install the CC4 version of the PI-API library.
You can re-install the version 5 library if needed.

Do you want to install the ANSI C++ (SC5) [Y] or the CC4 version [N]?
N<cr>
Version (5 5 1) API install type = 0
Installing Base System - PI API
API Installation Script
Setting Working Directory
Installing /opt/piapi/bin/pistart /opt/piapi/bin/pistop
```

```
Installing /opt/piapi/bin/sitestart
Installing /opt/piapi/bin/sitestop
Installing /opt/piapi/build/sitelink
Installing /opt/piapi/bin/apiverify
Installing /opt/piapi/bin/apiprocs
Installing /opt/piapi/lib/libpiapi.so
Installing /opt/piapi/lib/libpiapi.a
Installing files in /opt/piapi/bin:
  apisnap bufserv bufutil iorates ioshmcls ioshmsrv     isbuf mqcls mqmgr mqsrv
pilogsrv shootq
Running the Site Specific Link Script
#
```

After this, the PI API is installed in the directory `/opt/piapi`. The directory structure is as shown here:

```
total 16
-r--r--r--   1 root     545         1960 Sep  6  2000 PIAPIunix.txt
drwxr-xr-x   2 root     other        512 Mar 26 09:30 bin
drwxr-xr-x   3 root     545         1024 Mar 26 09:30 build
drwxr-xr-x   2 root     other        512 Mar 26 09:30 dat
-rw-r--r--   1 root     other       1085 Mar 26 09:30 install.log
drwxr-xr-x   2 root     other        512 Mar 26 09:30 lib
#
```

We can see a `bin` directory as well as the `lib` and `dat` directories added to the system.

5.6 Installing the PI AIS interface (fxbais).

In the previous section we installed the portion that takes care of the PI server communication. Next comes the AIS interface part. This takes care of the interaction between FoxAPI and the PI Client. The communication between the Foxboro API the PI client is established through the `fxbais` program and below are the steps to install the `fxbais` program files. Assuming the tar files are available in the `/opt/tmp` directory:

```
# cd /opt/tmp<cr>
# cp /cdrom/fxbais_tar.Z .<cr>
# uncompress fxbais_tar.Z<cr>
# cd /opt/piapi<cr>
# tar xvf /opt/tmp/fxbais_tar<cr>
```

This installs the interface files in the directory **`./interfaces/fxbais`**.
All the program files AND the “PIAPI to FoxAPI interface” are now in place and can be configured.

5.7 Configuring the PI Client.

We already set the PI Server node name during the previous steps. The Server name is set in a configuration file which is located in the `/opt/piapi/dat` directory. The PI Client configuration file is named **`piclient.ini`**. The default file with the server name in place is printed here:

[/opt/piapi/dat/piclient.ini]

```
[PISERVER]
PIHOMENODE=PISERVER
DSTMISMATCH=0
[TCP/IP]
PORT=5450
[APIBUFFER]
BUFFERING=0
```

Some of the settings are explained here:

5.7.1 The PISERVER (PIHOMENODE) setting

As we can see, the PISERVER is defined in this file (**PIHOMENODE=**). You can modify this file and stop/start PI to address the new server.

5.7.2 The TCP/IP port.

PI uses a specific TCP/IP port for its transport. The default port is 5450 which defined by the **PORT=** parameter in this file.

5.7.3 The buffering function.

PI allows the PI Client to buffer the collection data to be buffered on the local system in case communication with the PI Server fails. In order for this to work, the parameter **BUFFERING=** must be set to "**1**". The default is "**0**", no local buffering.

5.7.4 Rotating the PIMESSLOG file.

It may be necessary to "manage" the pimesslogfile. (PI creates one file per day so the space used may become a problem), to avoid this from happening you could create a cron clean-up entry as it was suggested by OSI documentation:

```
# PI create a new file at midnight every day
#  keep only the last 7 days:
20 10 * * * find /opt/piapi/dat \( -name "pimesslogfile.*" \) -mtime +7 -exec rm
-f {} \;
```

The entry above will cleanup the log files older than 7 days.

6 Configuring proper PI startup at boot.

It would be a nice thing if the PI software where started at boot time. In the OSI documentation this is covered but this NOT the way I like it. The documented Foxboro way is presented here.

6.1 Step 1: Editing / creating /etc/fox/user_apps.dat

Important:
In the directory **/usr/fox/bin** you will find a script named **fox_apps**. This script will have the location of the **user_apps.dat** file that will be processed. The location can be different from the example below.

The proper way to start "third party" applications is through a file that is read after a reboot of the system. The file required for this is found in the `/etc/fox` directory. The name of this file is `user_apps.dat` but this file does not exsist by default.
When this file does exsist it is read when all Foxboro related processes have been started (but may not have finished starting). When the file does not exist, it must be created with exactly that name:

[/etc/fox/user_apps.dat]
```
/opt/piapi/go_pistart
```

When the I/A system boots, it will (in due time) process the lines in this file and (in this example) tries to start a script `go_pistart` in the `/opt/piapi` directory. This takes us to the second step: The `go_pistart` script.

6.2 Step 2 : The go_pistart script.

This script is called through the `/etc/fox/user_apps.dat` file. This file is NOT the one provided by OSI. The one we are going to use is printed here:

[$PIHOME/go_pistart]
```
#!/bin/sh
# /opt/piapi/go_pistart
# script to initialize PI startup
# This script should be started from /etc/fox/user_apps.dat
# revision 1.4 / April 6, 2005 / R.Deen-Invensys
# Added "nohup" to pistart command.
# Added pistop to cleanup before pistart
# Added check for foxapi process
# In English now
PIHOME=/opt/piapi ; export PIHOME
cd $PIHOME/bin
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$PIHOME/lib ; export LD_LIBRARY_PATH
#echo $PIHOME
#echo $LD_LIBRARY_PATH
echo `pwd`
# Before starting PI, FoxAPI must be up and running
while [ `ps -ef|grep foxapi | grep -v grep |wc -l` -lt 4 ]
do
   echo "FoxAPI not running yet, wait 10 sec. and retry"
   sleep 10
done
echo "FoxAPI is running."
echo "Stopping the PI software for a clean start..."
$PIHOME/bin/pistop
sleep 10
```

```
echo "Starting the PI software..."
nohup $PIHOME/bin/pistart
```

This script does not require any user modifications under normal circumstances. It will check if the FoxAPI processes are up and running and when conditions are OK, continues with the OSI PI startup.
As we can see, the last line attempts to start the `pistart` program in `/opt/piapi/bin`. Let's go to the next step.

6.3 Step 3: The `/opt/piapi/bin/pistart` script.

The `pistart` script is an OSI supplied script that takes care of their software startup. The `pistart` script takes care of these basic PI functions:

- Start the buffering server
- Start the PI Message Log
- Start the I/O Rates Program
- Start the site specific Applications (sitestart)

The script is listed here for your reference:

[$PIHOME/bin/pistart] OSI supplied file

```
#!/bin/sh
################################################################################
# @(#)pistart  1.2  09/17/98
#
# 12-Jun-93  GZC  Original
# 15-Jul-93  RAB  Modified for Formal Release
# 03-Aug-93  JHP  Handle the different OS specific ps arguments
# 18-apr-94  jhp  added #!sh to force exec in bourne shell
# 26-apr-95  jhp  warning on user other than root rather than exit
# 03-Aug-95  hks  add grep -v grep for OSF platforms, look for
#                 piadmin as well as root
# 24-apr-96  hks  add starting of bufserv
# 22-nov-96 hks  add sleep after bufserv start.
# 28-aug-98  cah  Add check for USER and define with current user if needed
#                 and enclose in quotes in conditional.
# 17-sep-98  cah  Move message server before bufserv.
#
# This script file is used to start the PI system. It is intended for use
# under the Bourne or a related shell. The following tasks are executed:
#
#      * Start the buffering server
#      * Start the PI Message Log
#      * Start the I/O Rates Program
#      * Start the site specific Applications (sitestart)
#
################################################################################
echo "Starting PI..."
#
#  set ps arguments
PSARG=-e  # -cx for solaris 1
#
# Check for root account
#
if [ ${USER:-notdefined} = "notdefined" ]; then
   USER=`whoami`
fi
if [ "$USER" != "root" -a "$USER" != "piadmin" ]; then
   echo "Warning: USER is not "root" or "piadmin" applications may not start."
fi
```

```
#
# Verify the PIHOME Environment Variable
#
if [ ${PIHOME:-notdefined} = "notdefined" ]; then
        echo "The PIHOME environment variable has not been defined"
        echo "Please define PIHOME and run pistart again."
        echo "For example (from the Bourne shell):"
        echo "% PIHOME=/home/pi"
        echo "% export PIHOME"
        exit 1
fi

#
# Remember Current Directory and Change to $PIHOME/bin
#
CURRENTDIR=$PWD
cd $PIHOME/bin
#
#
# Start Applications
#
# Message Log

pid=`ps $PSARG | grep 'mqsrv' | grep -v grep | awk '{ print $1 }'`
if [ "$pid" != "" ]; then
   echo "The PI Message Log Server is already Running"
else
   echo "Starting the PI Message Log Server"
   ./mqsrv &
fi
pid=`ps $PSARG | grep 'mqmgr' | grep -v grep | awk '{ print $1 }'`
if [ "$pid" != "" ]; then
       echo "The PI Message Log Manager is already Running"
       echo "The PI Message Log Manager will be restarted"
       kill -9 $pid
fi
echo "Starting the PI Message Log Manager"
./mqmgr &

# Is buffering required
./isbuf
BUFFERING=$?
if [ $BUFFERING -eq 0 ]
then
    echo "Buffer server is not started."
else
  #    reset the shared memory and semaphores
  #    start the buffer server
  pid=`ps $PSARG | grep 'bufserv' | grep -v grep | awk '{ print $1 }'`
  if [ "$pid" != "" ]; then
     echo "The PI Buffer Server is already Running"
  else
      echo "Clearing buffer memory and locks."
      ./bufutil -u
     echo "Starting the PI Buffer Server"
     ./bufserv &
     sleep 5
  fi
fi

# I/O Rates
pid=`ps $PSARG | grep 'ioshmsrv' | grep -v grep | awk '{ print $1 }'`
if [ "$pid" != "" ]; then
   echo "The I/O Rates Shared Memeory Server is already Running"
else
   if [ -x ./ioshmsrv ]; then
      echo "Starting I/O Rates Shared Memory Server"
      ./ioshmsrv &
   fi
fi
```

```
pid=`ps $PSARG | grep 'iorates' | grep -v grep | awk '{ print $1 }'`
if [ "$pid" != "" ]; then
   echo "The I/O Rates Program is already running"
else
   if [ -x ./iorates ]; then
      echo "Starting the I/O Rates Program"
      ./iorates &
   fi
fi

# Site Specific Applications

if [ -x ./sitestart ]; then
       ./sitestart
fi

fi

#
# Write Startup Message to Log File
#
pid=`ps $PSARG | grep 'mqsrv' | grep -v grep | awk '{ print $1 }'`
if [ "$pid" != "" ]; then
   ./shootq "PI Start Completed"
else
   echo "Errors starting PI--Message Log Server not running"
fi

#
# Return to the starting directory
#
cd $CURRENTDIR
echo "PI Startup is Complete"
exit 0
```

When `pistart` is completed, the next file that is started is the `sitestart` script. This script starts the “site specific” options for PI.

6.4 Step 4: The `/opt/piapi/bin/sitestart` script.

The `/opt/piapi/bin/sitestart` script is where the PI interface scan classes are specified amongst other things. The `sitestart` script will perform some basic checks during processing but the most interesting part it in the end, again indicated in blue, you will find the line(s) that you should edit to match your requirements:

[$PIHOME/bin/sitestart] Modified OSI supplied file

```
################################################################################
# @(#)sitestart       1.2 09/05/95
# File: sitestart
#
# This file is provided to allow site specific applications to be started
# whenever the pistart is executed. An appropriate application stop script
# should be inserted into the sitestop file.
#
################################################################################
cd $PIHOME/interfaces/fxbais
$PIHOME/interfaces/fxbais/fxbais.sh

[$PIHOME/interfaces/fxbais/fxbais.sh] Modified OSI script
#!/bin/sh
# "@(#)fxbais.sh 1.3  00/02/10"
#
#  (C)Copyright OSI Software,Inc. San Leandro, California 1998
# Shell procedure to start 'fxbais' as a background process
#
```

```
#  Revision
#   1.0   3-18-98 cah  Original
#   1.1  12-16-98 cah  Added new command line options.
#   1.2  04-30-99 cah  Generalized for multiple interface copies with
#                      a numeric argument to this script.
#   1.3  02-10-00 cah  removed /sn by default;
#
# Change PROG_NAME if multiple copies of the interface are run.
if [ "${1:-undef}" = "undef" ]; then
  PROG_NAME=fxbais
else
  PROG_NAME=fxbais$1
fi
# if PIHOME not in environment,  then set for this procedure
if [ "${PIHOME:-undef}" = "undef" ]; then
  echo "PIHOME not defined"
  WORKDIR=$PWD
else
  WORKDIR=$PIHOME/dat
fi
echo "Output file is in $WORKDIR"
ISRUNNING=`ps -ef | grep "$PROG_NAME" | grep -v grep | grep -v $PROG_NAME.sh`
if [ "$ISRUNNING" = "" ]; then
  if [ -f $WORKDIR/${PROG_NAME}.out ]; then
     echo " Rename ${PROG_NAME}.out as ${PROG_NAME}.old"
     /bin/mv "$WORKDIR/${PROG_NAME}.out" "$WORKDIR/${PROG_NAME}.old"
  fi
#---
#  The input parameters are:
#   -ps=?      Point source character.                          [required]
#   -f=#[,#]   Scan classes in seconds and offsets.  [at least one required]
#   -id=#      Interface number (used in location1 definition)    [required]
#   -q         Queue input data before a putsnapshot call.     [recommended]
#   -stopstat  I/O timeout written to PI on normal shutdown.   [recommended]
#   -write     Required if outputs are desired.
#   -ec=#      Event counter number for inputs.
#   -ecout=#   Event counter number for outputs.
#   -ecuinp=#  Event counter number for unbuffered inputs.
#   -ecuout=#  Event counter number for unbuffered outputs.
#
#   -host=piserver:portid If the pi server is different from the default,
#              the -host flag may be used.
#   -sn        Snapshots, not exceptions on data received by interface.
#                (Interface does exceptions, not FoxAPI)          [optional]
#   -sio       Suppress initial output of current snapshot value in source
#                tag during point database loading.
#   -fdb=#     List of debug flags.
#   -perf=#    Hours between scan performance summary.
#
#---  Edit the following line  ---
  $PIHOME/interfaces/fxbais/$PROG_NAME -host=PISERVER:5450 -ps=H -id=1 -q -write
-stopstat \
  -ec=1 \
  -f=00:00:01 -f=00:00:05 \
      > $WORKDIR/${PROG_NAME}.out 2>&1 &
  echo " Starting interface ($PROG_NAME process)"
else
  echo " Interface ($PROG_NAME process) now running. Can not restart."
fi
exit 0
# end
```

7 Some diagnostics

It may be nice to know that you are not entirely on your own in this cruel world. There are some files that will provide some feedback on what is going on during startup.
Some places to look for clues:

- `/etc/fox/user_apps.log`
- `/opt/piapi/dat/pimesslogfile`

The status of all actions performed by `/etc/fox/user_apps.dat` are reflected tin the log file in the same directory. Here is an example of such a log file:

[/etc/fox/user_apps.log]

```
Thu Mar 25 08:19:23 GMT 2004
Starting user applications
--------------------------
--------------------------
/opt/piapi/bin
FoxAPI not running yet, wait 10 sec. and retry
FoxAPI not running yet, wait 10 sec. and retry
FoxAPI not running yet, wait 10 sec. and retry
FoxAPI not running yet, wait 10 sec. and retry
FoxAPI not running yet, wait 10 sec. and retry
Stopping the PI software...
Stopping PI...
 Stopping Foxboro I/A AIS interface (PI part)
   Can take about 5 sec per i/f list to finish
msgq_client: No such file or directory
25-Mar-04 08:20:14
root: fxastop> Stopping Foxboro I/A interface

 fxbais interface not running.
msgq_client: No such file or directory
25-Mar-04 08:20:14
root: fxastop> fxbais interface not running.

The I/O Rates Program was NOT Found
The I/O Rates Shared Memory Server was NOT Found
The PI Buffer Server was NOT Found
The PI Message Log Server was NOT Found
The PI Message Log Manager was NOT Found
PI Shutdown is Complete
FoxAPI is running.
Starting the PI software...
Starting PI...
Starting the PI Message Log Server
Starting the PI Message Log Manager
Clearing buffer memory and locks.
Starting the PI Buffer Server
Starting I/O Rates Shared Memory Server
Starting the I/O Rates Program
Output file is in /opt/piapi/dat
 Rename fxbais.out as fxbais.old
 Starting interface (fxbais process)
PI Startup is Complete
/opt/piapi/go_pistart successful
--------------------------
--------------------------
/usr/fox/bin/fox_apps: test: argument expected
All listed user applications processed
```

Another file is the `/opt/piapi/dat/pimesslogfile`. An example of this file is found here:

[$PIHOME/dat/pimesslogfile]

```
...
...
...
23-Mar-04 13:27:32
FXBIA- 1> (UTC time on sever node - UTC time on interface node) = -37
23-Mar-04 13:27:33
iorates>PI-API> Initial connection to [PISERVER][-1]
23-Mar-04 13:27:33
iorates>getiotags> 0 rate tag(s) registered.
23-Mar-04 13:28:58
apisnap>PI-API> Initial connection to [PISERVER][-1]
23-Mar-04 13:37:19
root: PI Shutdown Started
23-Mar-04 13:37:19
root: fxastop> Stopping Foxboro I/A interface
23-Mar-04 13:39:08
root: fxastop> Stopping Foxboro I/A interface
23-Mar-04 13:39:41
root: fxastop> Stopping Foxboro I/A interface
23-Mar-04 13:45:57
root: fxastop> Stopping Foxboro I/A interface
23-Mar-04 13:45:58
FXBIA- 1> Closing all data sets
23-Mar-04 13:45:58
FXBIA- 1> Closed 1 data sets
23-Mar-04 13:46:03
root: fxastop> fxbais interface successfully stopped.
23-Mar-04 13:46:14
bufserv>PI-API> APIBUFFER: Starting buffer server.
23-Mar-04 13:46:19
bufserv>PI-API> Initial connection to [PISERVER][-1]
23-Mar-04 13:46:20
root: PI Start Completed
23-Mar-04 13:46:24
iorates>PI-API> Initial connection to [PISERVER][-1]
23-Mar-04 13:46:24
iorates>getiotags> 0 rate tag(s) registered.
23-Mar-04 13:46:48
root: PI Shutdown Started
23-Mar-04 13:46:48
iorates>> Process exiting.
23-Mar-04 13:46:49
bufserv>PI-API> APIBUFFER: Buffer server exiting.
23-Mar-04 13:46:59
bufserv>PI-API> APIBUFFER: Starting buffer server.
23-Mar-04 13:47:03
bufserv>PI-API> Initial connection to [PISERVER][-1]
23-Mar-04 13:47:04
FXBIA-> /opt/piapi/interfaces/fxbais/fxbais -host=Piserver:5450 -ps=H -id=1 -q
-write -stopstat -ec=1 -f=00:00:01 -f=00:00:05
23-Mar-04 13:47:04
FXBIA- 1> Starting interface, Point source:  H
23-Mar-04 13:47:04
FXBIA- 1> Uniint version>@(#)uniint.cxx     3.2.6
23-Mar-04 13:47:04
FXBIA- 1> API version> 1.3.4
23-Mar-04 13:47:04
```

An indication that PI is running properly may be the **`fxbais`** process. This is running on your AP/AW when PI is active:

[Some active processes on Foxboro hosts]

```
...
...
root  1694  1568  0   Mar 25 ?        0:00 smon_strh 61SM02 SM0N01
root  1857     1  0   Mar 25 ?        0:00 /opt/piapi/interfaces/fxbais/fxbais
-host=PISERVER:5450 -ps=H -id=1 -q -write
```

```
root  1988     1  0   Mar 25 ?        0:00 /opt/aim/bin/histsend
root  1759     1  0   Mar 25 ?        0:00 /opt/fox/ais/bin/foxapi wrproc
root  1761     1  0   Mar 25 ?        0:00 /opt/fox/ais/bin/foxapi om_poll
root  1849     1  0   Mar 25 ?        0:00 ./iorates
root  1827     1  0   Mar 25 ?        0:00 ./mqmgr
root  1822     1  0   Mar 25 ?        0:02 ./mqsrv
root  1906     1  0   Mar 25 ?        0:00 /usr/lib/dmi/dmispd
root  1834     1  0   Mar 25 ?        0:00 ./bufserv
root  1844     1  0   Mar 25 ?        0:00 ./ioshmsrv
root  1950     1  0   Mar 25 ?        0:00 /opt/aim/bin/apimgr wrproc
```

u