Tobie Fehlaber

Britehouse®
A DIVISION OF DIMENSION DATA

OpenEdge REST– Switching from Classic Appserver to PAS

PUG africa

Agenda

Britehouse
A DIVISION OF DIMENSION DATA

- What is PAS
- What is the difference between Classic Appserver and PAS
- Why change ?
- Properties
- Logging
- Operating modes and Session Types
- Transports replace adapters
- Broker/Agent architectures
- Scaling the server
- ABL Sessions
- Steps to switch from Classic Appserver to PAS
- Deploying REST service to PAS
- DEMO

01

PAS

Progress Application Server - PAS

The Progress Application Server for OpenEdge is an Apache Tomcat Web server and servlet container that is customized to provide application server support for OpenEdge applications and to respond to HTTP/HTTPS client requests.

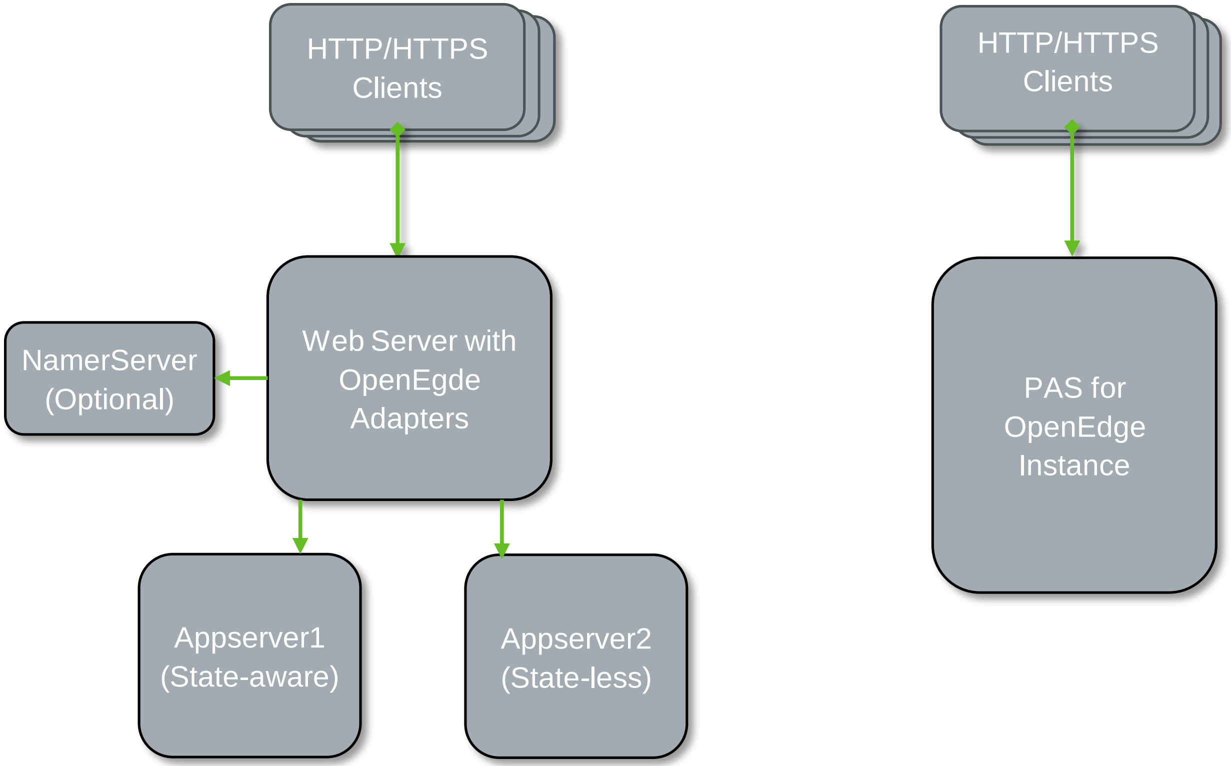
Difference between Classic Appserver and PAS?

- The OpenEdge AppServer must be configured with an external Web server with adapters (AIA, REST, SOAP) in order to respond to HTTP/HTTPS client requests.
- Requires NameServer
- With OpenEdge AppServer, you must run a separate instance for each session model and operational mode that you need to support.

- PAS for OpenEdge responds directly to HTTP/HTTPS requests because it is a Web server with a default ROOT application configured to support APSV (analogous to AIA), REST, and SOAP requests.
- Client requests handled by PAS for OpenEdge are not managed and routed by a NameServer process.
- A single PAS for OpenEdge instance supports both session-managed and session-free connection models

Difference between Classic Appserver and PAS?

Britehouse
A DIVISION OF DIMENSION DATA



Why change ?

If you want to use OpenEdge 12 you have to !

Properties:

- PAS for OE does not use a ubroker.properties file to hold configuration information. There is no shared properties file that can be referenced by multiple PAS server instances.
- Each instance of PAS for OE has its own /conf directory that contains multiple configuration files, including **openedge.propertes**.
- The **openedge.propertes** file is similar in format to ubroker.properties. It contains some of the same properties as ubroker.properties , and some PAS-specific properties.
- In addition to **openedge.propertes**, an PAS for OpenEdge instance's /conf directory contains the following properties files:
    - ❑ *appserver.properties* — Specifies Java properties used by PAS for OE and Web applications. See /conf/appserver.propertes.README for more information.
    - ❑ *catalina.properties* — Specifies Java system properties used by the TCMAN administration utilities and by deployed Web applications. See /conf/catalina.propertes.README for more information.
    - ❑ *jvm.properties* — Contains a list of JVM startup command line options.

Logging:

- Log files for the Progress Application Server for OpenEdge are saved, by default, in an instance's /logs directory.
- In the Classic OpenEdge AppServer, logging is enabled in the ubroker.properties file.
- In PAS for OE, you enable logging, set logging levels, and specify the name logging files in the following properties files:
  - ❖ $CATALINA_BASE/conf/logging.properties for Tomcat
  - ❖ $CATALINA_BASE/conf/openedge.properties for the PAS for OpenEdge Multi-session Agent
  - ❖ $CATALINA_BASE/webapps/webapp_name/WEB-INF/logging.xml for Web applications

Operating modes and Session Types :

- The Classic OpenEdge AppServer is configured to run in one of four operating modes (state-reset, state-aware, stateless, and state-free). *Essentially, each mode specifies how a client is bound to an agent and whether context information is maintained from one client session to the next. An OpenEdge AppServer can only run in a single operating mode. You must run a separate OpenEdge AppServer for each operating mode that you need to support.*
- Progress Application Server for OpenEdge is simpler. Instead of a variety of operating modes, there are two session types, session managed and session free.
  - Session managed in Progress Application Server for OpenEdge means that a client is bound to a single session. *It is somewhat similar to the state-reset and state-aware modes in the OpenEdge AppServer with respect to the fact that context is maintained between client requests. Subsequent client requests are handled in the context of the same session.*
  - Session free in Progress Application Server for OpenEdge means that a client is not bound to a single session. Subsequent requests from a client can be handled in different sessions.

Also, be aware that a PAS for OE instance can respond to both session managed and session free client requests.

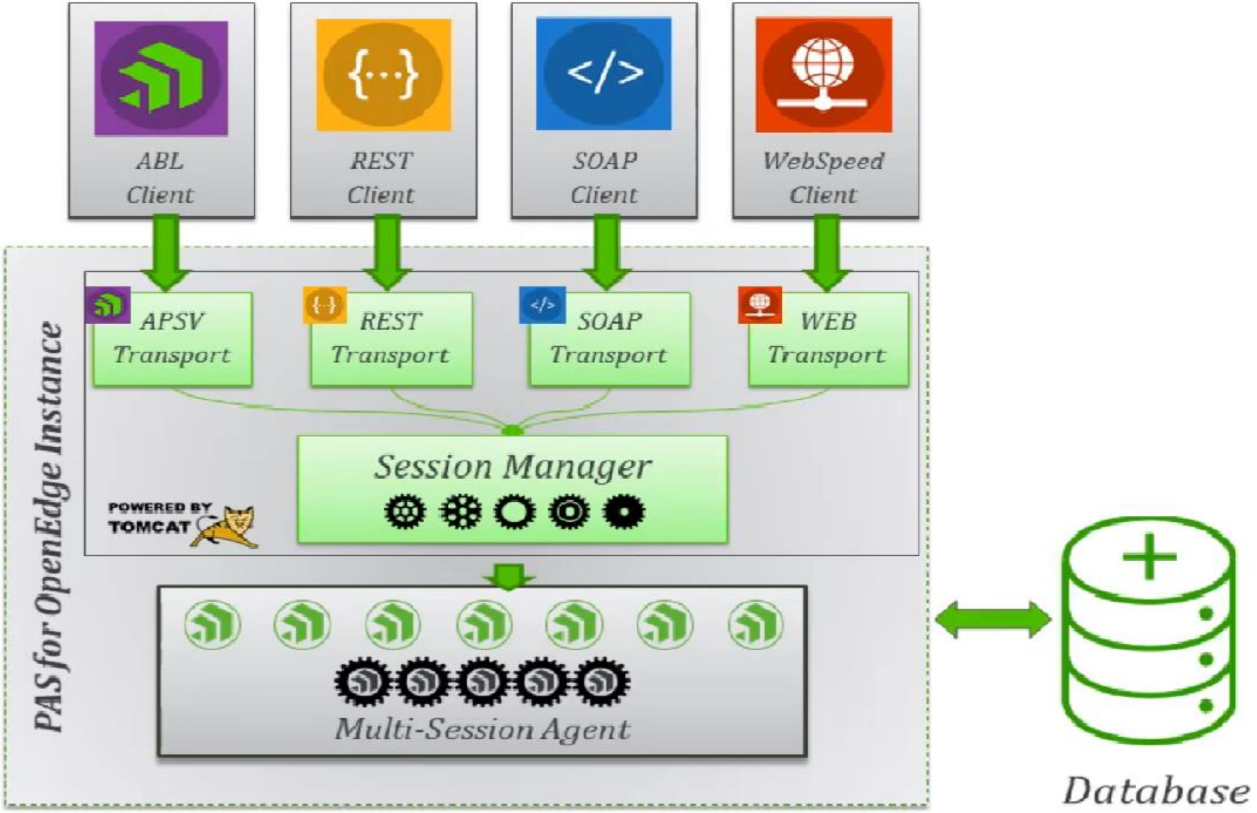
Transports replace adapters:

- To provide HTTP/HTTPS client access to Web services associated with a classic OpenEdge AppServer, you must install WSA (SOAP) or REST adapters on a separate Web server. HTTP/HTTPS client access to OpenEdge applications that are not deployed as Web services requires the installation of an AIA adapter on a separate Web server.
- In Progress Application Server for OpenEdge, support for HTTP/HTTPS Web services and OpenEdge applications is included in an instance's default /webapps/ROOT application. The root application contains transports that support SOAP, REST, WEB, and APSV (equivalent to AIA) access. Since PAS for OpenEdge is a Web server, there is no need to install and configure adapters on a separate Web server.

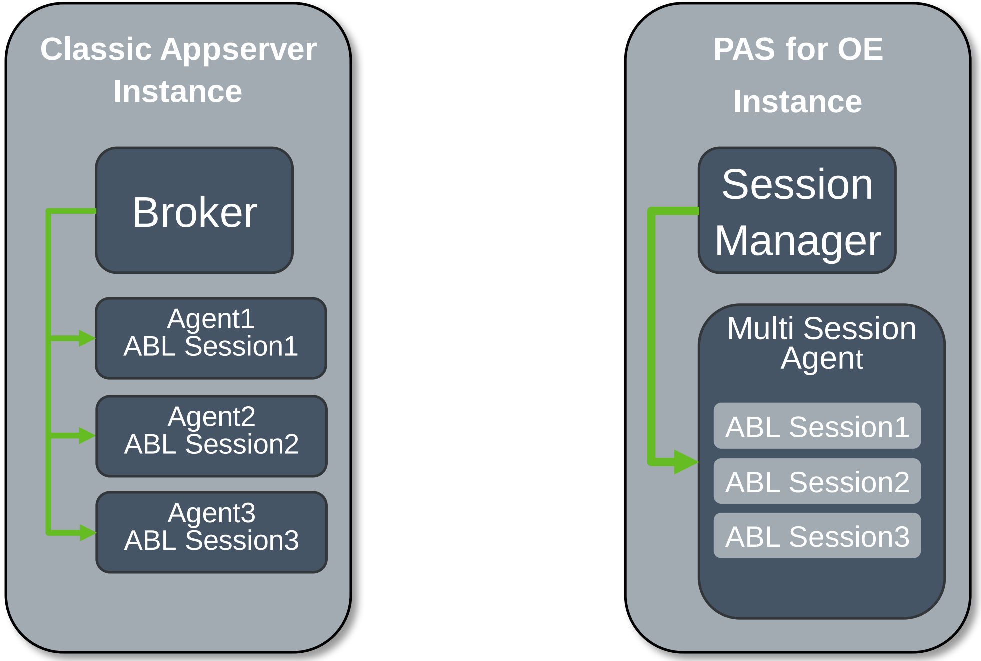
Each transport, or protocol, is accessed via a default URL path in an instance. The defaults (which you can change) are listed in the following table:

Transport	URL path	Notes
APSV	/apsv	Supports the OpenEdge AppServer protocol over HTTP
REST	/rest	Supports REST RPC over CFX and Camel
SOAP	/soap	Support SOAP 1,1
WEB	/web	Supports WebSpeed applications

Transports replace adapters:



Broker/Agent architectures:



Scaling the server:

- Both the Classic AppServer and the Progress Application Server are scalable—they can be configured to optimize performance as the workload increases. Note, the differences in how you scale each type of server in the following areas:
  - **Adding and trimming agents** — The first step in scaling a server is to adjust the number of agents in order to maximize the server's response time under a given workload.
  - In both the Classic AppServer and PAS, you can edit a properties file or use OEE/OEM to adjust the number of agents running on a server. However, PAS also allows you to modify the number of sessions that an agent supports. An agent in the Classic AppServer is a single session agent, whereas PAS agents are multi-session agents.
  - **Load balancing** — The second step in scaling is load balancing. Load balancing basically involves adding more instances of the server and configuring an environment where the workload is share among those instances.
  - With the Classic AppServer, you can register multiple AppServer instances that support the same application services using an OpenEdge NameServer. You assign a weight factor to determine how the NameServer balances the load (client requests) among instances.
  - You can create multiple server instances, but there is no NameServer in PAS. Instead of a NameServer, load balancing is achieved in PAS for OE through an Apache extension module or through third party applications. For example, see:
  - http://httpd.apache.org/docs/current/mod/mod_proxy.html

ABL Sessions:

- In the Classic AppServer, each agent is an OS process that handles a single ABL session.
- In PAS, an agent is a single OS process that can run multiple ABL sessions. It is often referred to as a Multi-session Agent (MSAgent).
- The ABL Session Manager in PAS is not identical to, but is functionally similar to the broker in the Classic AppServer. It responds to client requests by communicating with an agent process to assign a session, to make sure that ABL code executes, and to ensure that a response is returned to the client. If there are no free sessions in one agent, it launches another agent.
- Note: You configure properties in a PAS instance's /conf/openedge.properties file. It is similar in format to the ubroker.properties file, but it only applies to a single instance. All individual PAS for OpenEdge instances have their own openedge.properties file.
- Unlike the OpenEdge AppServer, you do not need to run a separate instance of PAS in a mode to support a particular session model. A single instance of PAS supports both session-managed and session-free connections.
- The client request determines the mode of the session. Sessions running in different modes can run concurrently on a single instance of PAS.

ABL Sessions (Cont.):

- In PAS, you must specify separate startup/shutdown procedures for agents and sessions.
- In the Classic AppServer, the *srvrStartupProc* and *srvrShutdownProc* procedures execute on both agent and session start/stop events because the Classic AppServer agent is bound to a single session.
- In PAS, you specify startup/shutdown procedures in an instance's /conf/openedge.properties file.
- *sessionStartupProc* and *sessionShutdownProc* properties to specify the procedures that execute on ABL session start/stop events.
- *agentStartupProc* and *agentShutdownProc* properties to specify the procedures that execute on agent start/stop events.

Steps to switch from Classic Appserver to PAS:

Configure PAS for OpenEdge

- Create an Instance
- Merge properties files
- Deploy new ABL Web Application
- Enable REST Transport

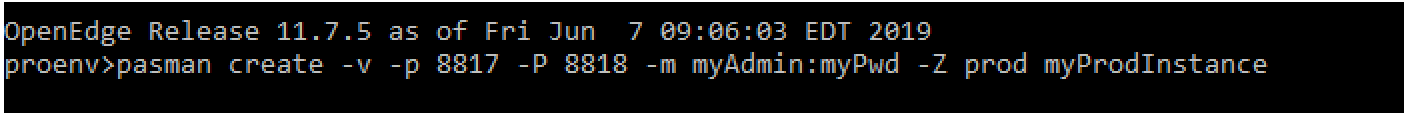
Migrate Classic Apsserver Code

- Move server code to the instance
- Extract the .paar file
- Deploy the .paar file

Test Service connection

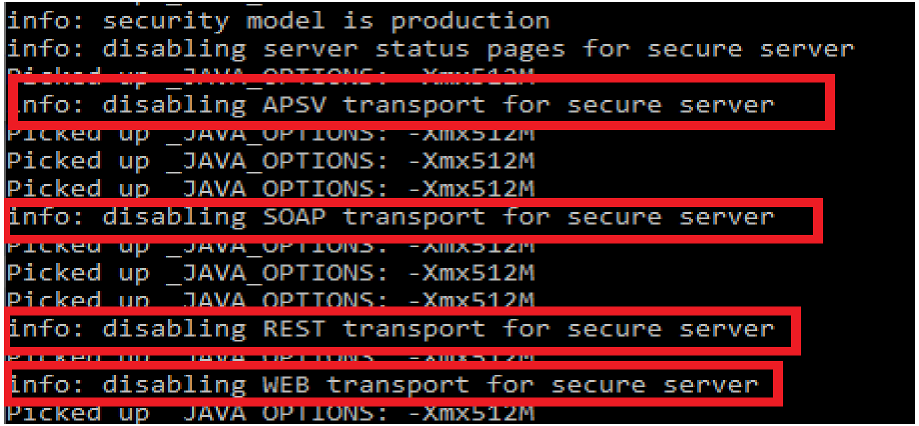
- Start the instance
- Test connection in a browser
- Stop the instance

How do I create a PAS Server:



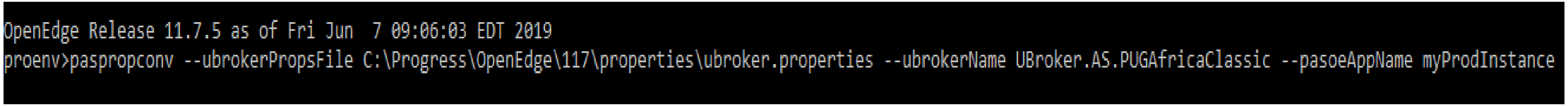
```
OpenEdge Release 11.7.5 as of Fri Jun  7 09:06:03 EDT 2019
proenv>pasman create -v -p 8817 -P 8818 -m myAdmin:myPwd -Z prod myProdInstance
```

- pasman create : to create pas server instance in proenv
- -v : verbose option to provide feedback as the insance is created
- -p : tcp port that listens for http messages
- -P : tcp port that listens for https messages
- -m :username and password to replace the tomcat:tomcat default username and password used by apache
- -Z prod : used to create the production instance , this locks down the server with production server security settings
- Instance_pathname : if no path is specified it will be created in the current path.



```
info: security model is production
info: disabling server status pages for secure server
Picked up _JAVA_OPTIONS: -Xmx512M
info: disabling APSV transport for secure server
Picked up _JAVA_OPTIONS: -Xmx512M
Picked up _JAVA_OPTIONS: -Xmx512M
Picked up _JAVA_OPTIONS: -Xmx512M
info: disabling SOAP transport for secure server
Picked up _JAVA_OPTIONS: -Xmx512M
Picked up _JAVA_OPTIONS: -Xmx512M
Picked up _JAVA_OPTIONS: -Xmx512M
info: disabling REST transport for secure server
Picked up _JAVA_OPTIONS: -Xmx512M
info: disabling WEB transport for secure server
Picked up _JAVA_OPTIONS: -Xmx512M
```

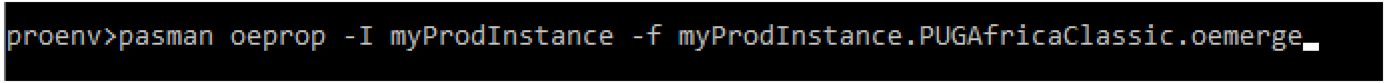
Merge Properties files:



```
OpenEdge Release 11.7.5 as of Fri Jun  7 09:06:03 EDT 2019
proenv>paspropconv --ubrokerPropsFile C:\Progress\OpenEdge\117\properties\ubroker.properties --ubrokerName UBroker.AS.PUGAfricaClassic --pasoeAppName myProdInstance
```

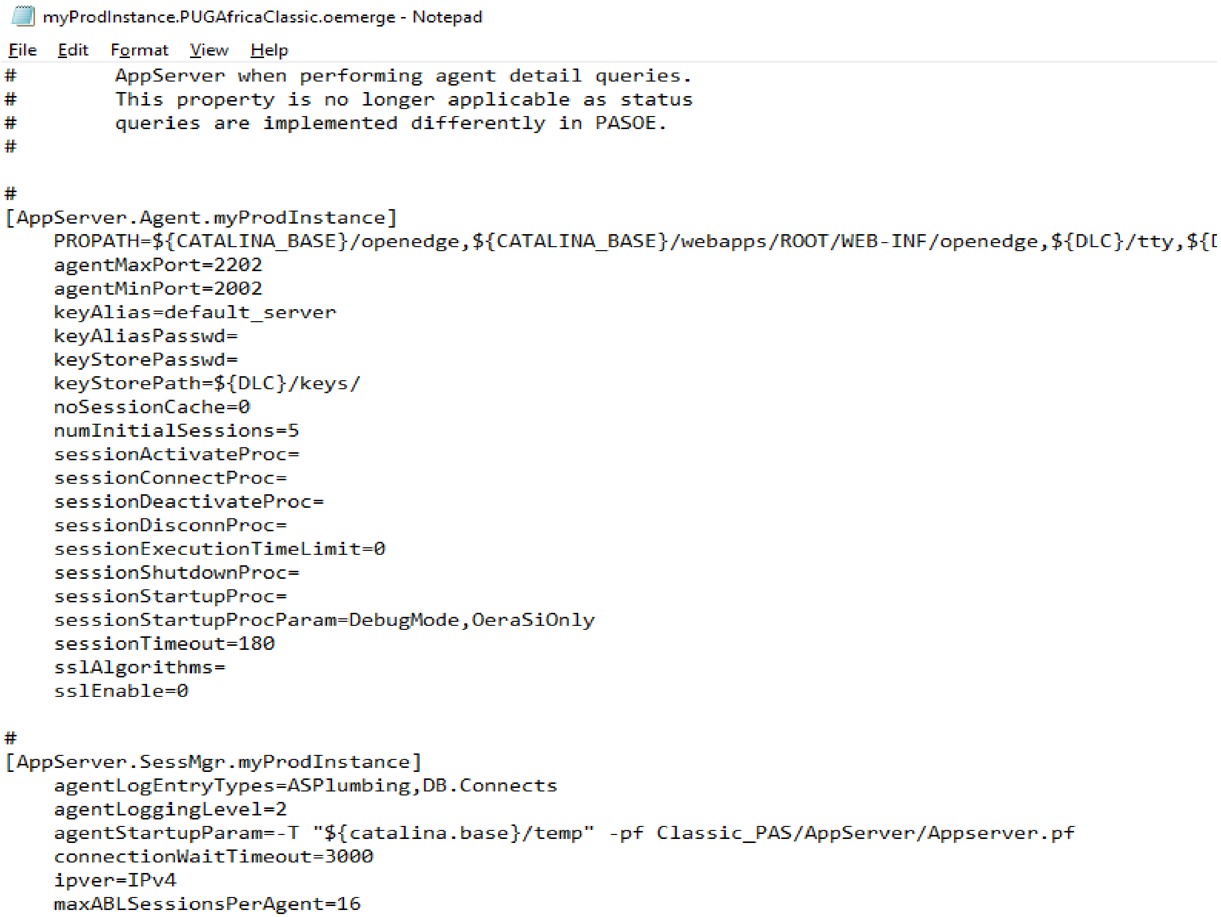
- Convert existing properties to a temporary merge file with following script.
- **paspropconv** utility uses the following options :

  --ubrokerPropsFile C:\Progress\OpenEdge\117\properties\ubroker.properties
  (Path to the current ubroker properties file)
  --ubrokerName UBroker.AS.PUGAfricaClassic
  (Fully qualified broker name of the broker you are converting)
  --pasoeAppName myProdInstance (Name of the new Instance)
- This creates a merge file of converted values free for you to review and customize before merging them into your openedge.properties files.
- File is called myProdInstance.PugClassic.oemerge file that you can customize.
- NOTE : This is only available from OpenEdge 11.7.3
- REMEMBER at this stage you can enable APSV,REST,SOAP and WEB transports.
- [myProdInstance.ROOT.APSV]
  adapterEnabled=1
- To Apply changes :



```
proenv>pasman oeprop -I myProdInstance -f myProdInstance.PUGAfricaClassic.oemerge
```

Merge Properties files:


myProdInstance.PUGAfricaClassic.oemerge - Notepad
File Edit Format View Help

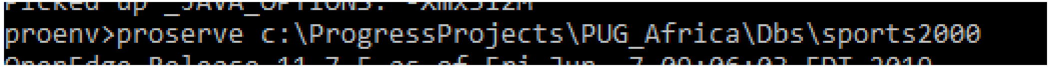
```
#        AppServer when performing agent detail queries.
#        This property is no longer applicable as status
#        queries are implemented differently in PASOE.
#

#
[AppServer.Agent.myProdInstance]
    PROPATH=${CATALINA_BASE}/openedge,${CATALINA_BASE}/webapps/ROOT/WEB-INF/openedge,${DLC}/tty,${D
    agentMaxPort=2202
    agentMinPort=2002
    keyAlias=default_server
    keyAliasPasswd=
    keyStorePasswd=
    keyStorePath=${DLC}/keys/
    noSessionCache=0
    numInitialSessions=5
    sessionActivateProc=
    sessionConnectProc=
    sessionDeactivateProc=
    sessionDisconnProc=
    sessionExecutionTimeLimit=0
    sessionShutdownProc=
    sessionStartupProc=
    sessionStartupProcParam=DebugMode,OeraSiOnly
    sessionTimeout=180
    sslAlgorithms=
    sslEnable=0

#
[AppServer.SessMgr.myProdInstance]
    agentLogEntryTypes=ASPlumbing,DB.Connects
    agentLoggingLevel=2
    agentStartupParam=-T "${catalina.base}/temp" -pf Classic_PAS/AppServer/Appserver.pf
    connectionWaitTimeout=3000
    ipver=IPv4
    maxABLSessionsPerAgent=16
```

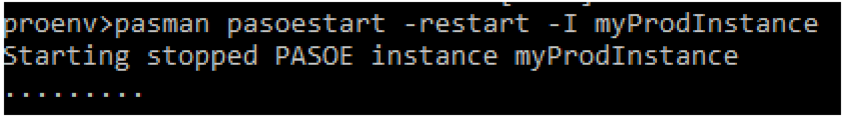
Start PAS and Test application:

- Inspect openedge.properties file and make sure everything is fine.
- Check that database is up and running :



```
proenv>proserve c:\ProgressProjects\PUG_Africa\Dbs\sports2000
```

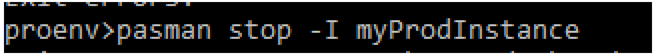
- Start PAS :



```
proenv>pasman pasoestart -restart -I myProdInstance
Starting stopped PASOE instance myProdInstance
.........
```

- If you get errors trying to start check agent log files in “myProdInstance/logs”

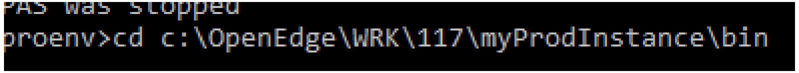
- Connecting ABL code to PAS server is different to classic :
  - Classic Connect : cConnect = “-S 5164 –H machinename -AppService asbroker1”.
  - PAS connect : cConnect = "-URL http://machinename:8817/apsv -sessionModel Session-Free".
- Stop PAS instance :



```
proenv>pasman stop -I myProdInstance
```

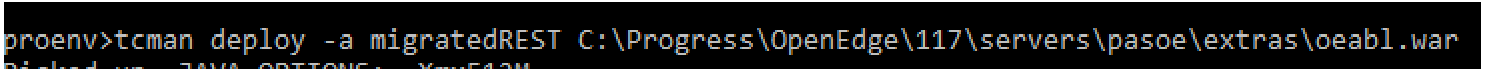
Deploy new ABL Web Application , Enable REST Transport:

- Go to PAS instance :



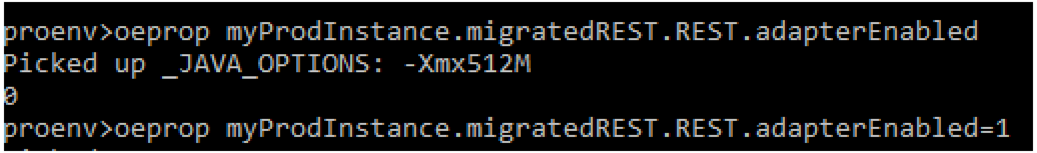
```
PAS was stopped
proenv>cd c:\OpenEdge\WRK\117\myProdInstance\bin
```

- Deploy new Web Application :



```
proenv>tcman deploy -a migratedREST C:\Progress\OpenEdge\117\servers\pasoe\extras\oeabl.war
Picked up _JAVA_OPTIONS: -Xmx512M
```

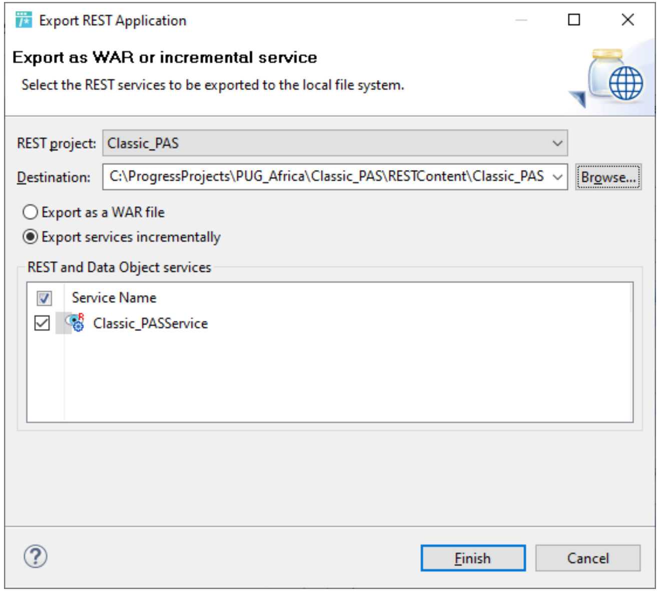
- Enable REST Transport :



```
proenv>oeprop myProdInstance.migratedREST.REST.adapterEnabled
Picked up _JAVA_OPTIONS: -Xmx512M
0
proenv>oeprop myProdInstance.migratedREST.REST.adapterEnabled=1
```

Extract and Deploy .paar file :

- Extract .paar file :



Export as WAR or incremental service

Select the REST services to be exported to the local file system.

REST project: Classic_PAS

Destination: C:\ProgressProjects\PUG_Africa\Classic_PAS\RESTContent\Classic_PAS

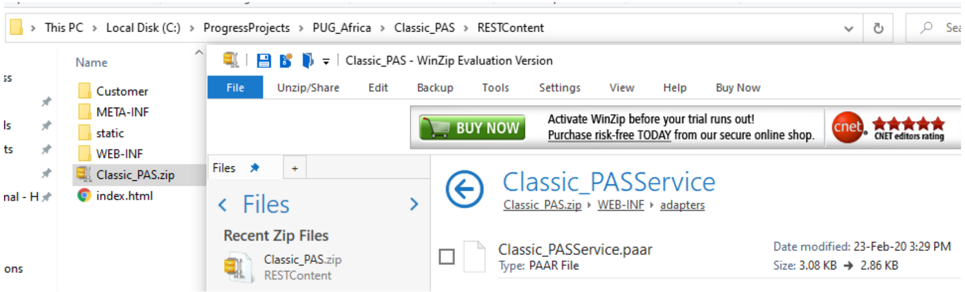
○ Export as a WAR file

● Export services incrementally

REST and Data Object services

☑ Service Name

☑ Classic_PASService



This PC › Local Disk (C:) › ProgressProjects › PUG_Africa › Classic_PAS › RESTContent

Name
Customer
META-INF
static
WEB-INF
Classic_PAS.zip
index.html

Classic_PAS - WinZip Evaluation Version

Activate WinZip before your trial runs out!
Purchase risk-free TODAY from our secure online shop.

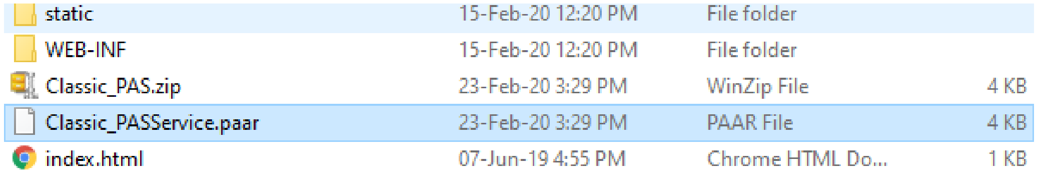
Recent Zip Files

Classic_PAS.zip
RESTContent

Classic_PASService

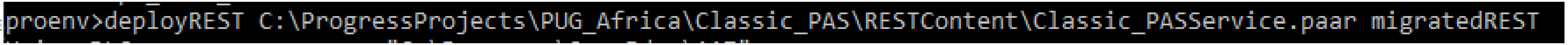
Classic_PAS.zip › WEB-INF › adapters

□ Classic_PASService.paar
Type: PAAR File
Date modified: 23-Feb-20 3:29 PM
Size: 3.08 KB → 2.86 KB



static	15-Feb-20 12:20 PM	File folder	
WEB-INF	15-Feb-20 12:20 PM	File folder	
Classic_PAS.zip	23-Feb-20 3:29 PM	WinZip File	4 KB
Classic_PASService.paar	23-Feb-20 3:29 PM	PAAR File	4 KB
index.html	07-Jun-19 4:55 PM	Chrome HTML Do...	1 KB

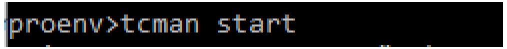
- Deploy .paar file :



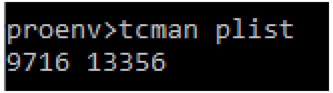
```
proenv>deployREST C:\ProgressProjects\PUG_Africa\Classic_PAS\RESTContent\Classic_PASService.paar migratedREST
```

Test Service connection:

- Start Instance :

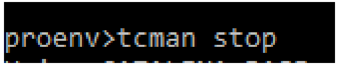


```
proenv>tcman start
```

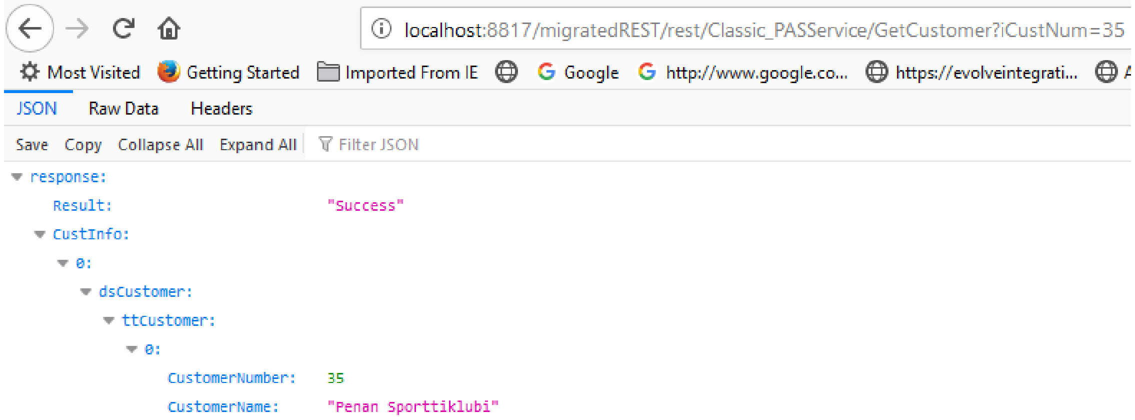


```
proenv>tcman plist
9716 13356
```

- Test connection in web browser :
- REST Service URL format – hostname:port/webapp_name/rest/service_name/resource_name
- Example REST service: http://localhost:8817/migratedREST/rest/Classic_PASService/GetCustomer?iCustNum=35
- Stop Instance :



```
proenv>tcman stop
```



localhost:8817/migratedREST/rest/Classic_PASService/GetCustomer?iCustNum=35

```
▼ response:
    Result:                    "Success"
  ▼ CustInfo:
    ▼ 0:
      ▼ dsCustomer:
        ▼ ttCustomer:
          ▼ 0:
              CustomerNumber:  35
              CustomerName:    "Penan Sporttiklubi"
```

Demo

```
1. pasman create -v -p 8821 -P 8822 -m admin:admin -Z prod PUGAfrica
2. paspropconv --ubrokerPropsFile C:\Progress\OpenEdge\117\properties\ubroker.properties
   --ubrokerName UBroker.AS.PUGAfricaClassic --pasoeAppName PUGAfrica
3. pasman oeprop -I PUGAfrica -f PUGAfrica.PUGAfricaClassic.oemerge
5. pasman pasoestart -restart -I PUGAfrica
6. tcman deploy -a PUGAfricaREST c:\Progress\OpenEdge\117\servers\pasoe\extras\oeabl.war
7. oeprop PUGAfrica.PUGAfricaREST.REST.adapterEnabled=1
8. deployREST C:\ProgressProjects\PUG_Africa\Classic_PAS\RESTContent\Classic_PASService.paar PUGAfricaREST
9. tcman start
10. tcman plist
11. tcman env
12. openedge.properties - [PUGAfrica.ROOT.REST]
                          adapterEnabled=1
13. tcman stop
14. pasman stop -I PUGAfrica
```