Navigating Pacific

A Voyage through an OpenEdge AppServer Hackathon

A Hearty Crew of Pacific Pirates

PROGRESS EXCHANGE 2014

Chart

Destination	Duration
An introduction to PAS for OpenEdge	
Learning labs	
1. Creating and starting an Instance	
2. Client connections	
3. Configuring an Instance for your application	
4. Configuring Event Procedures	
5. Connecting your databases	
6. Deploying services	
7. Administration & monitoring	
Hack ahoy!	

An Introduction to PAS for OpenEdge

PROGRESS EXCHANGE 2014

Architectural Drivers

- **Secure** production web server
  - Installation, administration
- **Simpler**
  - Administration, scalability, application migration, deployment
  - AppServer connection and operating states
- **Customer Extensible**
  - Open REST APIs for customer developed metrics, monitoring, and administration
  - Installation tailoring
- **Better analysis tools**
  - Built-in metrics gathering, current state queries
- **Faster and optimizes resources**
  - Runs same ABL application and client load with less memory and CPU consumption

Pacific Application Server for OpenEdge (PAS for OE)

- Installable web server product that merges into a single package the functionality of:
  - Tomcat
  - OpenEdge AppServer
  - AppServer Adapters
- Runs in Progress's unified Pacific Application Server (PAS) platform
  - Same web server installed with Rollbase-private installation and with Corticon
  - Common server administration and configuration functionality
- Supported on only 64 bit platforms (Linux, Solaris, HPUX-IA, AIX, Windows 2008/2012)
- Two products: Development server and Production server

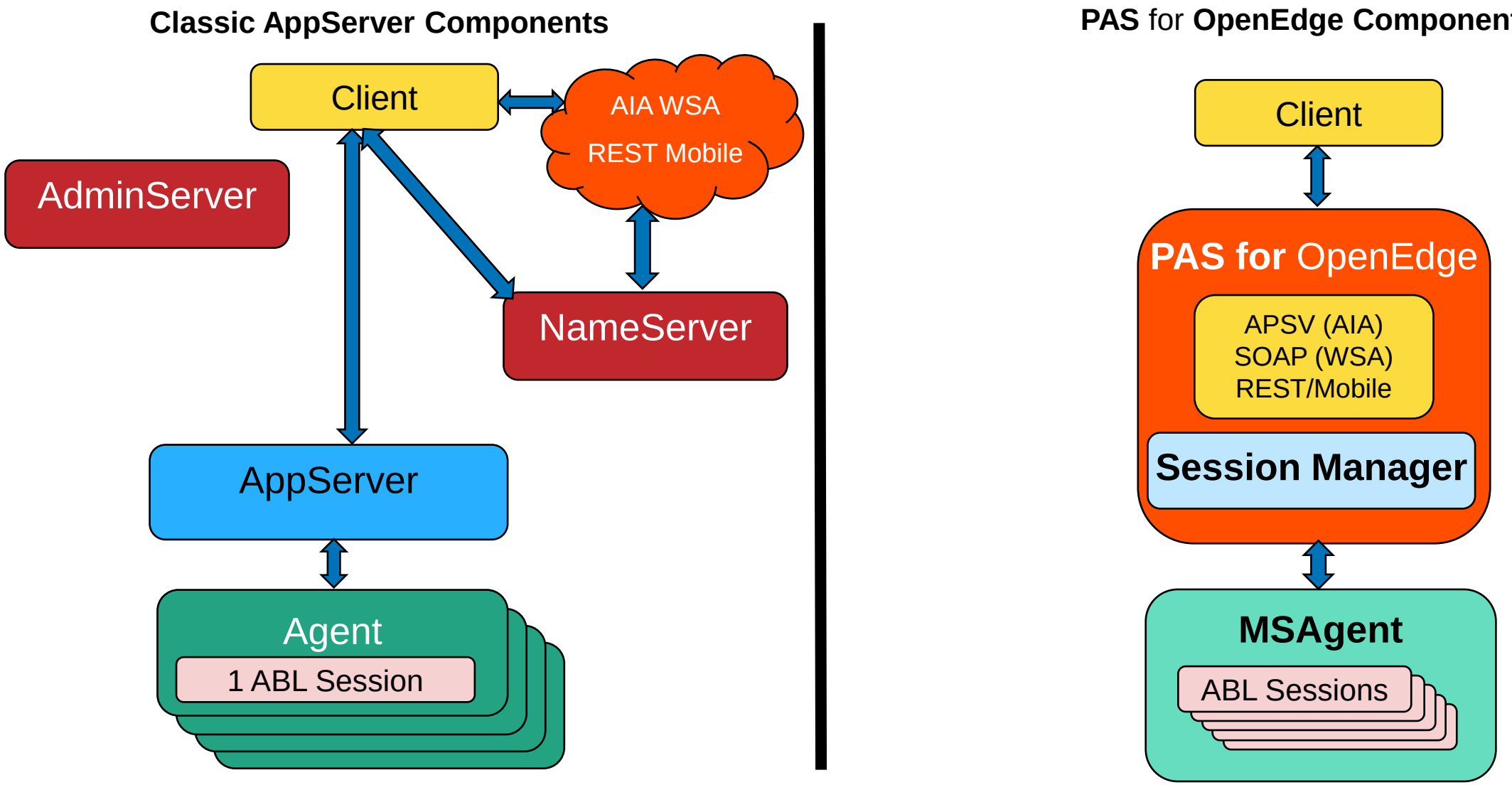
Pacific Application Server Platform

- Created from Apache Tomcat 7.0.55 distribution
- Configured for production by default
  - Default ROOT application replaced
  - Tomcat manager web applications not installed
  - Auto deployment disabled by default
  - Shutdown port disabled on Unix
  - JMX not enabled by default
  - Web crawler filtering enabled
- Spring Security Framework included
- Realms and roles defined to implement access control
- Enhanced command line tool to configure and manage server
- Will be default implementation for Rollbase and Corticon

PAS for OE Architectural Concepts and Terms

- **PAS** is a Java web application server with Progress extensions
  - Optional web application for remote administration using text, html, or JMX proxy APIs
- ***ABL services for Pacific Application Server (ABLPAS)***
  - An collection of Java web applications and resources installed and into a **PAS**
  - Java web applications use multi-session agent OS process (**MSAgent**)
  - *Optional* web application for remote administration using REST APIs
  - ***ABLPAS*** web services can be installed into the **PAS** of other PSC product installations
- **PAS for OE** is an OpenEdge installed **PAS** with a pre-installed **ABLPAS** services
- You design, package, deploy, configure, debug, and control access to your ABL application in the context of a web application running in a web server

Architecture: Components



How Is PAS for OE Different to AppServer?

- **What's new?**
  - SESSION:CLIENT-TYPE = MULTI-SESSION-AGENT
  - Built-in Spring security for ABL (APSV) and SOAP client connections
  - agentStartup, -Shutdown event procedure (in addition to session-)
  - Set environment variables via setenv.[bat|sh]
  - CURRENT-REQUEST-INFO new attributes
  - Command-line and other tools
- **What's different?**
  - Client determines session management, not server
  - ABL AppServer `CONNECT()` arguments
    - Must use **`-URL`**
  - Both connect/disconnect and activate/deactive run for session-managed
  - Property names changed, more of them
  - `ubroker.properties` → `$CATALINA_BASE/conf/openedge.properties`
  - `SESSION:SERVER-OPERATING-MODE`
- **What's gone?**
  - Native protocol – everything's over HTTP on `/apsv`
  - AIA
  - The term broker
  - Name Server & NS load balancer
  - The Environment section in `ubroker.properties`
  - `-AppService CONNECT()` argument

Why Is This All Good for Me?

- Get more done with less code
- AppServer: session-managed binds user session & connection OR unbound connections and DIY session management
- PAS for OE: Session managed code (less code) without connection binding
  - Multi-session agent runs up to 7x* faster;
  - Your application's performance improvements will vary
- Standard deployment configuration with Tomcat

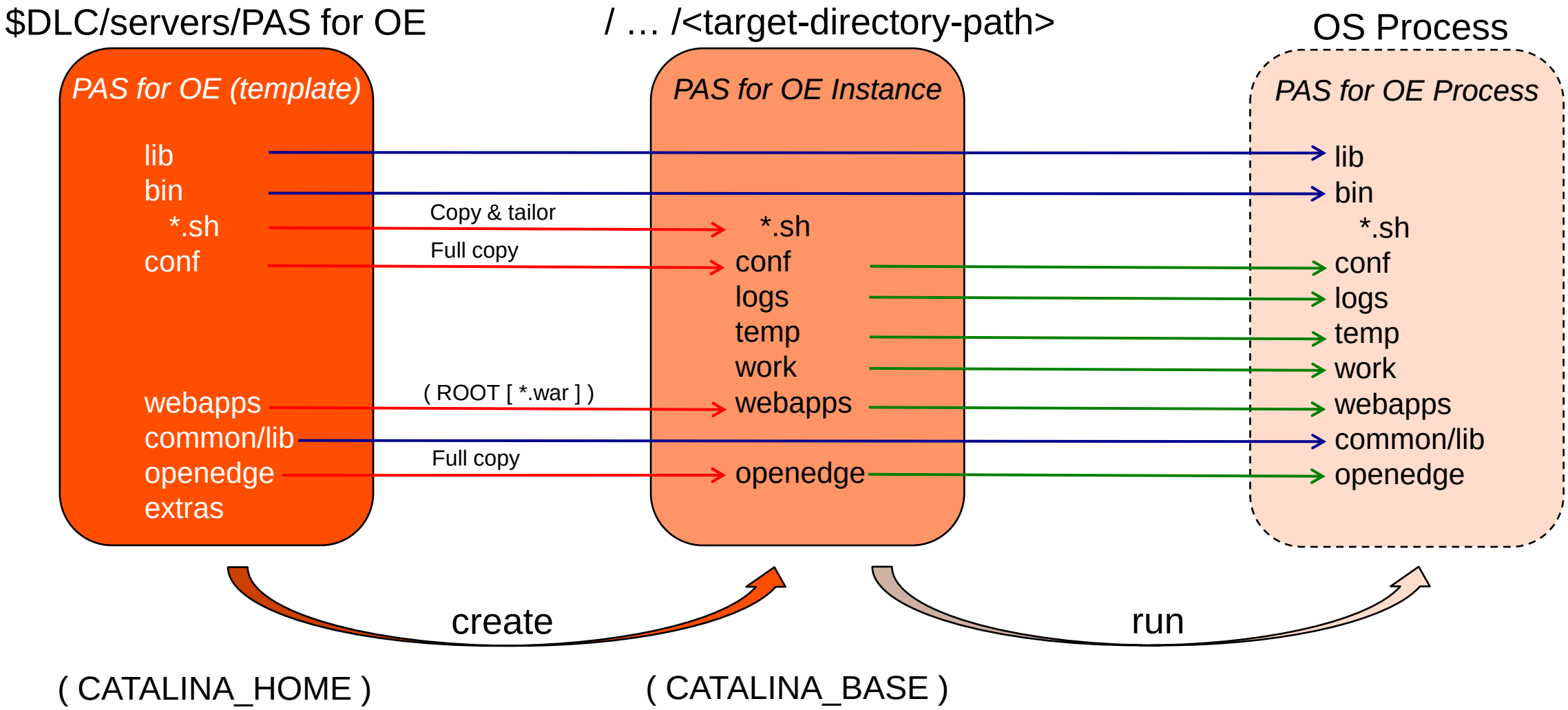
* Standard disclaimer applies

Creating and Starting an Instance

1

PROGRESS EXCHANGE 2014

Understanding PAS for OE Instance Run-time



PAS for OE Instances

- Each instance has a name
  - **oepas1** is the equivalent to AppServer's **asbroker1** default
- Instances have private file space outside the OE install
- An instance lives beyond OE uninstall
- Instances are transferrable between OE installs
- An instance can be packaged and redeployed

Command Line Tools for Creating and Managing Instances

- tcman.[sh/bat] primary tool for configuring PAS
  - $CATALINA_HOME/bin/tcman.bat to create instance
  - $CATALINA_BASE/bin/tcman.bat to configure instance
  - tcman.bat help <command> for usage info
  - Doesn't need proenv

Labs Preamble

- You should already have received login credentials for an Arcade instance (no relation)
  - You should ideally have installed some or all of your application on it

- For this hackathon, some paths

DLC = c:\Progress\OpenEdge115 ← OpenEdge location

CATALINA_HOME = c:\Progress\OpenEdge115\servers\pasoe ← Tomcat location

CATALINA_BASE = c:\hackathon\oepas1 ← Instance location

- All relative paths are relative to CATALINA_BASE unless otherwise specified
- The various scripts being run in this lab are the Windows variants (*.bat); Unixy variants (*.sh) are provided on those platforms

Learning Lab 1

- Start the preinstalled PAS instance
- Verify that it's running
- Stop the instance
- Clean up the log files

- Create a new instance (you'll need it later)
- Start/verify/stop your new instance

Everyone should, at this point, be able to create, start and stop an instance

Client Connections

2

PROGRESS EXCHANGE 2014

PAS for OE ABL Application Architecture Models

- ABL client controls session model via CONNECT method's *sessionModel* option
- Session-Free
  - Client's request can be executed in any ABL session, in any Agent, in any PAS for OE server
  - A client can execute concurrent ABL requests
  - User context managed entirely by the server application and client code
- Session-Managed
  - Required if you want to implement State-Reset, State-Aware, Stateless modes
  - Each client's request routed to same PAS for OE server, MSAgent, and ABL session
  - A user's context is stored within an Agent's ABL session between requests
  - Supports Automatic Transactions [that span multiple requests]
  - A client can execute pipelined async requests [to single ABL session]

Connecting to PAS for OE

- All connections now use HTTP(S)
  … even ABL clients
- Session management mode
  - Default is Session-Free
  - Selected when clients connect
- Use standard ABL to connect
  - Must use –URL
- Once connected, business logic unchanged
- Client connection parameters
  - -AppService becomes **-URL**
  - /wsa becomes **/soap**
  - /rest stays **/rest** (no change)

```
define variable AppHandle as handle.
define variable system-path as character.
define variable db-connections as character.

create server AppHandle.

AppHandle:connect("-URL http://localhost:8810/ROOT/apsv
                   -sessionModel Session-free ").

run testserver.p on AppHandle
      (output system-path, output db-connections).

message
'PROPATH= ' system-path skip
'DBs= ' db-connections
view-as alert-box info.

finally:
   AppHandle:disconnect() no-error.
   delete object AppHandle no-error.
end finally.
```

Service URLs

-URL http://localhost:8810/ROOT/apsv

instance service transport

GET http://localhost:8810/ROOT/rest/CustSvc/Customer

instance service transport resources

- ROOT is the default service. Ideally you should use something more meaningful named
- Services are the where you apply web server-level security
- URLs should always contain instance **and** service **and** transport
- Resources are optional, depending on transport

Learning Lab 2

- Start PDSOE
- Write an ABL procedure that
  - Connects to a running Instance, and
  - Runs a program on that Instance

- All the sample code is in `c:/hackathon/samples`

Configuring an Instance for Your Application

3

PROGRESS EXCHANGE 2014

PAS for OpenEdge configuration files

- Each instance's config files are in $CATALINA_BASE/conf
- Properties in multiple property files
  - Tomcat, PAS: eg logging.properties, appserver.properties
  - OpenEdge: openedge.properties file
- Core Pacific AppServer uses Java style property files
  - Managed with tcman config
  - tcman test
- Environment variables defined in instance's $CATALINA_BASE/bin in <app-name>_setenv.bat

Configuring PAS for OE

CATALINA_BASE = c:\hackathon\oepas1

- OpenEdge extension use ubroker style property file
  - $CATALINA_BASE/conf/openedge.properties
  - $CATALINA_BASE/bin/oeprop.bat manages openedge.properties

- Properties renamed and moved into different/new groups
  - Properties related to Agent process startup prefixed with agent
  - Properties related to Sessions prefixed with session
  - srvrStartupParam -> agentStartupParam
  - srvrStartupProc -> agentStartupProc
  - srvrStartupProcParam -> agentStartupProcParam

Where Does My Code Live?

- No changes to PROPATH required
- Production server: r-code **only**
- Potential r-code locations
  - As part of the instance
    - $CATALINA_BASE/openedge
    - Package and deploy instance as single unit
  - As part of an ABL web application
    - $CATALINA_BASE/webapps/ROOT/web-inf
    - Package and deploy individual services as war file
  - Whereever it lives today
    - Whatever you do today

```
[AppServer.Agent.oepas1]
uuid=http://localhost:8810/oepas1
PROPATH=${CATALINA_BASE}/openedge,${DLC}/tty


[AppServer.SessMgr.oepas1]
agentLogEntryTypes=ASPlumbing,DB.Connects
agentLogFile=
      ${catalina.base}/logs/oepas1.agent.log

agentLoggingLevel=3
publishDir=${CATALINA_BASE}/openedge
```

Learning Lab 3

- Trick lab … we will do a lot of this later

Configuring Event Procedures

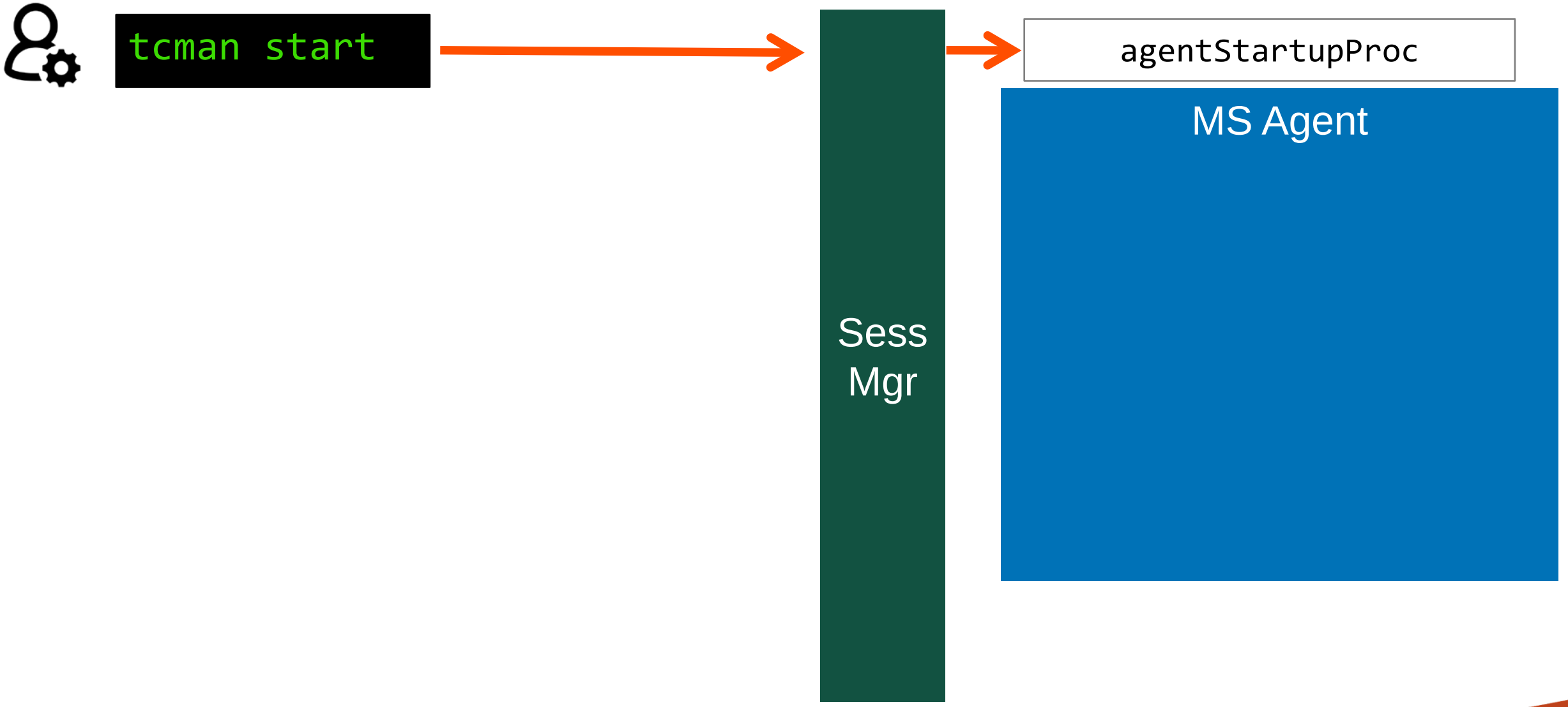
4

PROGRESS EXCHANGE 2014

Configuring Event Procedures

- Agent event procedures
  - Related to Agent process startup and shutdown
  - agentStartupProc and agentShutdownProc
    - Execute once when starting / stopping an MSAgent
- Session event procedures
  - Related to Session use
  - Sessions are state-aware and reusable
    - Support stateful and stateless requests
    - Must be cleaned up when done
- sessionStartupProc & sessionShutdownProc
  - Executes in PAS for OE when the MSAgent starts/stops each ABL SESSION
- sessionConnectProc & sessionDisconnectProc
  - Runs for all **Session-Managed** client connections
- sessionActivateProc & sessionDeactivateProc
  - Runs for (nearly, see later) all requests

Event Procedures



Event Procedures

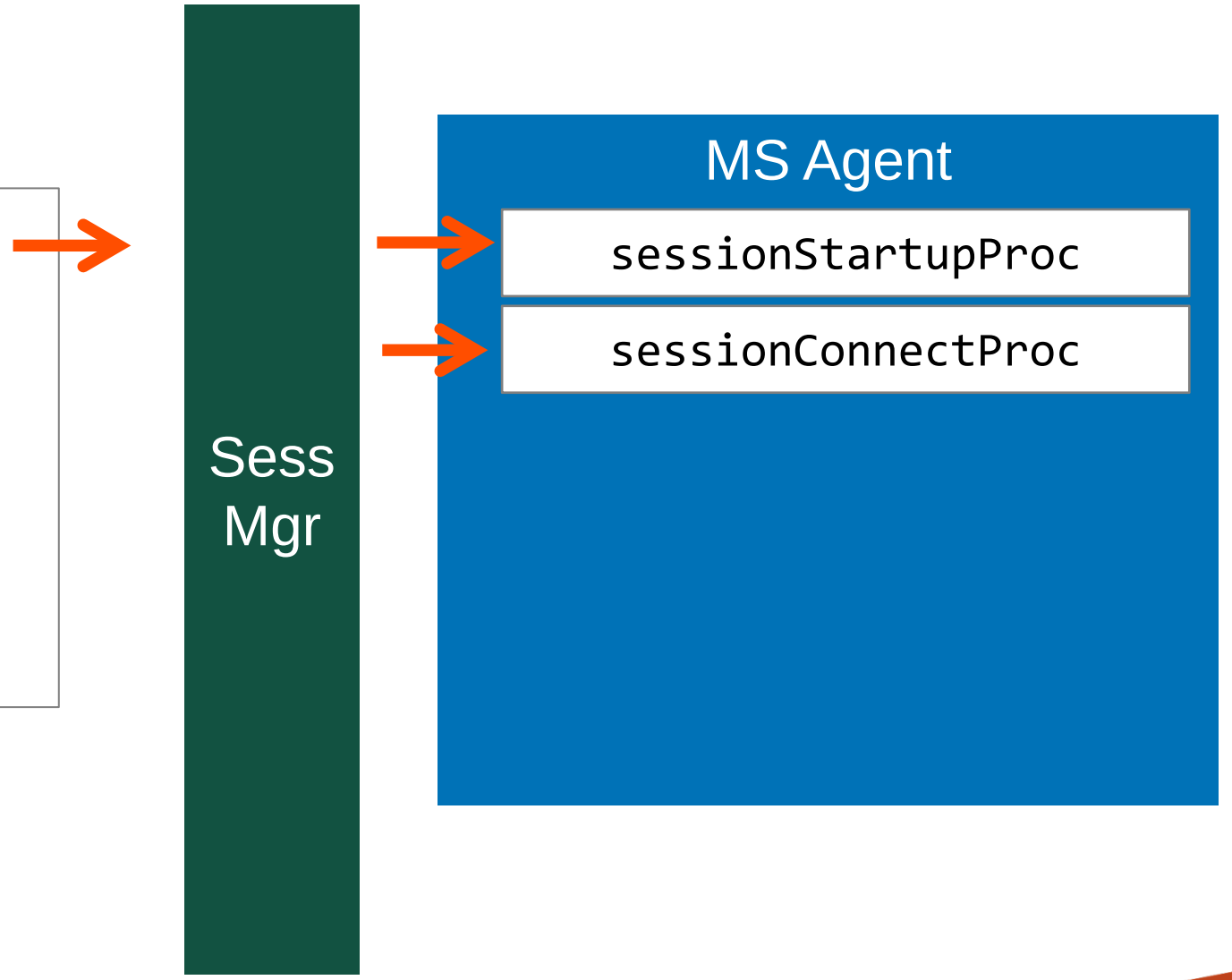
```
tcman start
```

```
AppHandle:connect("-URL
http://localhost:8810/apsv

-sessionModel Session-Managed").


run testserver.p on AppHandle.


AppHandle:disconnect().
```



Event Procedures

```
tcman start
```

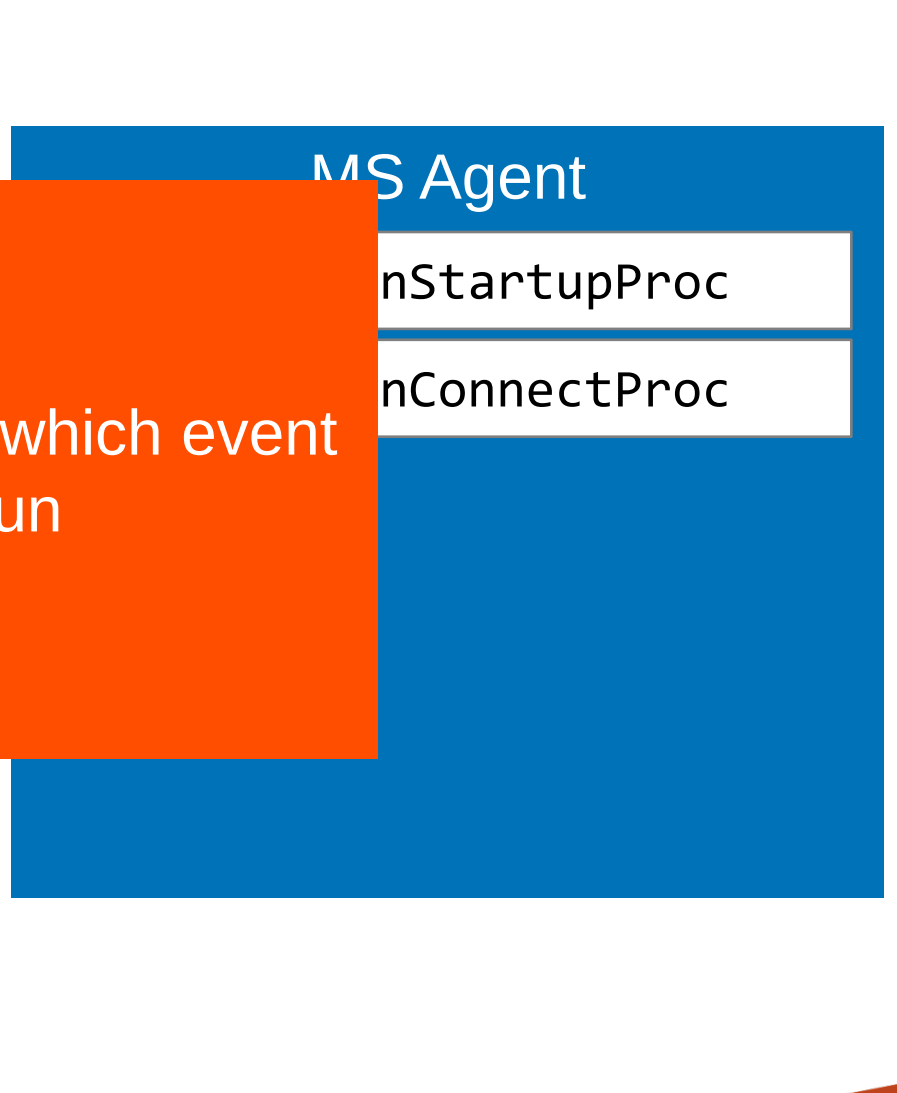
```
AppHandle:connect("-URL
http://localhost:8810/apsv

-sessionModel Session-Mana

run testserver.p on AppHandle

AppHandle:disconnect().
```

Session model affects which event procedures run



Event Procedures

```
tcman start
```

```
AppHandle:connect("-URL
http://localhost:8810/apsv

-sessionModel Session-Managed").


run testserver.p on AppHandle.


AppHandle:disconnect().
```

Sess Mgr

MS Agent

sessionActivateProc

Event Procedures

```
tcman start
```

```
AppHandle:connect("-URL
http://localhost:8810/apsv

-sessionModel Session-Managed").


run testserver.p on AppHandle.


AppHandle:disconnect().
```

Sess Mgr

MS Agent

```
sessionDeactivateProc
```

Event Procedures

```
tcman start
```

```
AppHandle:connect("-URL
http://localhost:8810/apsv

-sessionModel Session-Managed").


run testserver.p on AppHandle.


AppHandle:disconnect().
```

Sess Mgr

MS Agent

```
sessionDisconnectProc
```

Event Procedures

```
tcman start
```

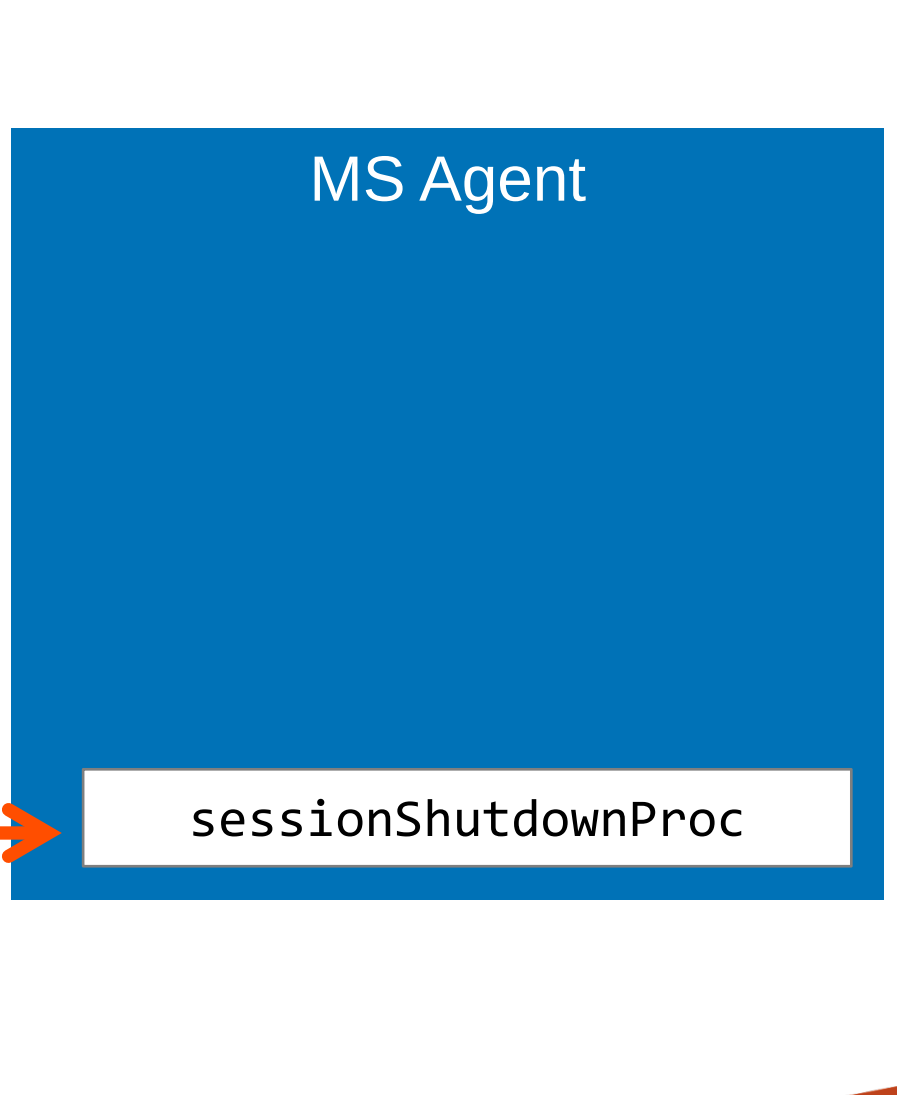
```
AppHandle:connect("-URL
http://localhost:8810/apsv

-sessionModel Session-Managed").


run testserver.p on AppHandle.


AppHandle:disconnect().
```

```
DELETE http://localhost:8810/oemanager
/applications/oepas1/sessions/123
```



Event Procedures

```
tcman start
```

```
AppHandle:connect("-URL
http://localhost:8810/apsv

-sessionModel Session-Managed").


run testserver.p on AppHandle.


AppHandle:disconnect().
```

```
DELETE http://localhost:8810/oemanager
/applications/oepas1/sessions/123
```

```
tcman stop
```

Sess Mgr

MS Agent

```
agentShutdownProc
```

Event Procedures

```
tcman start
```

```
AppHandle:connect("-URL
http://localhost:8810/apsv

-sessionModel Session-Managed").


run testserver.p on AppHandle.


AppHandle:disconnect().
```

```
DELETE http://localhost:8810/oemanager
/applications/oepas1/sessions/123
```

```
tcman stop
```

Sess Mgr

agentStartupProc

MS Agent

- sessionStartupProc
- sessionConnectProc
- sessionActivateProc
- sessionDeactivateProc
- sessionDisconnectProc
- sessionShutdownProc

agentShutdownProc

Event Procedures

```
tcman start
```

```
AppHandle:connect("-URL
http://localhost:8810/apsv

-sessionModel Session-Managed").


run testserver.p on AppHandle.


AppHandle:disconnect().
```

Event Procedures

```
tcman start
```

```
AppHandle:connect("-URL
http://localhost:8810/apsv
-sessionModel Session-Managed").


run testserver.p on AppHandle.


AppHandle:disconnect().
```



```
AppHandle:connect("-URL
http://localhost:8810/apsv
-sessionModel Session-Managed").
```

Event Procedures

```
tcman start
```

```
AppHandle:connect("-URL
http://localhost:8810/apsv
-sessionModel Session-Managed").


run testserver.p on AppHandle.


AppHandle:disconnect().
```

```
AppHandle:connect("-URL
http://localhost:8810/apsv
-sessionModel Session-Managed").
```

Sess Mgr

MS Agent

agentStartupProc

MS Agent

sessionStartupProc

sessionConnectProc

Configuring Event Procedures for Session Management

- Migrating Classic **State-Reset** AppServer Mode
  - Call **QUIT** ABL as last line – ideally in `FINALLY` block – of disconnect .p
  - ***AND*** also the below
- Migrating Classic **State-Aware** AppServer Mode
  - To achieve client **bound** behavior, the first statement in connect .p must be

    ```
    SESSION:SERVER-CONNECTION-BOUND-REQUEST = TRUE
    ```

  - And the (second-) last statement before disconnect .p exits must be

    ```
    SESSION:SERVER-CONNECTION-BOUND-REQUEST = FALSE
    ```

  - This fully disables the execution of Activate and Deactivate .p's and still leave them available for use by Session-Free client connections.

Configuring Event Procedures for Session Management

- Migrating Classic **Stateless** AppServer Mode
  - No changes required to event procedures.

- Migrating Classic **State-Free** AppServer Mode
  - No changes required to event procedures.

Connecting Your Databases

5

PROGRESS EXCHANGE 2014

Database Connections in PAS for OE

- Almost nothing changes
  - Connections via –pf or –db
  - ABL code the same
- Multi-session agent requires a dedicated DB connection
  - Used to manage all of that agent's sessions' DB connections
  - See it in PROMON as PASA (for agent) and PASN (for session)
  - Disconnect PASA at your own peril. Your Mileage Will Not Vary: it will be `0`.

Learning Lab 5

- Check that the Sports2000
- Add the database connection to the Instance properties
  - Add startup parameters to the startup properties

    OR

  - Add ABL code to an event procedure
- Run ABL client code to verify the database connection

Deploying Services

6

PROGRESS EXCHANGE 2014

Deploying Services

- SOAP Services
  - wsm file created by ProxyGen
  - Supports wsm files created with any version of OpenEdge
  - deploySOAP.bat deploys wsm file to running AppServer
- REST Services
  - paar file created by Progress Developers Studio
  - Extract paar file from war/zip created by PDS
  - Copy paar files from deployed 11.x REST applications
  - deployREST.bat deploys paar file to AppServer
  - Requires server restart if running
- Mobile Services (Apps)
  - Mobile apps (UI) and services (REST) can run together in same Instance

Learning Lab 6

- Deploy REST application RESTService
- Restart PAS for OE to get the service enabled
  - Archive log files on cleanup
- Test that the REST resources are available

- Deploy the SOAP application
- Run ABL test code written against the WSDL

Administration and Monitoring

7

PROGRESS EXCHANGE 2014

Configuring Logging

CATALINA_BASE = c:\hackathon\oepas1

- Logs can be found in $CATALINA_BASE/logs
- Adapter/Session Manager log – oepas1.[DATE].log
  - Logs roll over daily
  - Configured through $CATALINA_BASE/webapps/oepas1/web-inf/logging.xml
- Agent log – oepas1.agent.log
  - Configured in openedge.properties
  - No changes from classic AppServer other than property name prefix
  - agentLogEntryTypes supported
- tcman clean [-A]
  - Truncates/deletes log files
  - -A archives logs in timestamped subdirectory

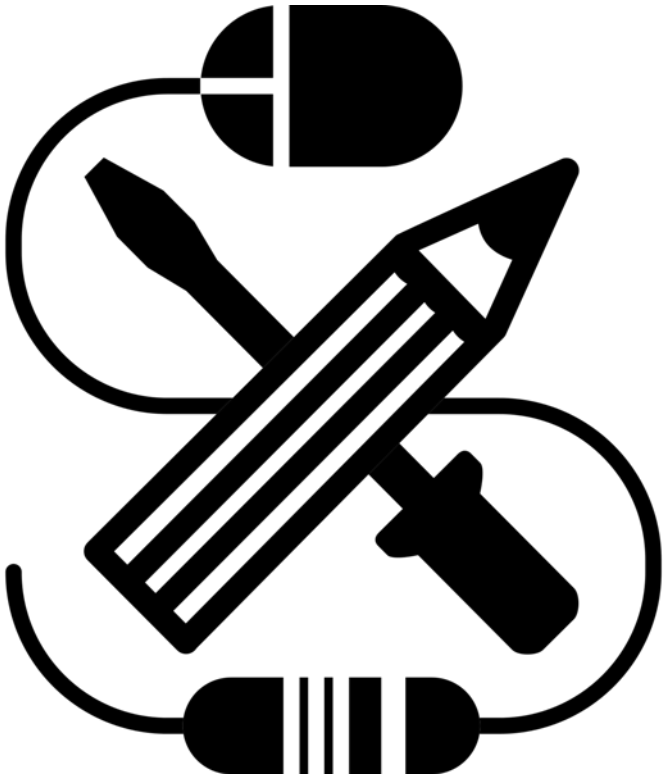
Administration and Monitoring

- Tomcat Manager
  - Deployed with development server
  - Provides basic server operation information
- JMX Support
  - Tomcat provides rich set of JMX Beans
  - Disabled by default – use tcman to enable
  - Not recommended for production servers – security concern
- OE Manager
  - Deployed with development server
  - Provides OE Specific JMX Beans
  - Provides REST API used by OEM

Learning Lab 7

- Increase logging levels
- Enable REST API
- Turning on JMX

Your Turn Now …Time to Hack



PROGRESS EXCHANGE 2014

Migrating Your Application

- You should have installed your application's bits by now. If not, there's a version of AutoEdge you can use (located at `c:/hackathon/autoedge`)

- You'll probably need to change the following (at a minimum)
  - PROPATH
  - Event procedures
  - ABL client connections
  - DB connections

- You may want to play with
  - Handing session-managed and –free modes in code
  - Securing access to the `/aspv` transport in `$CATALINA_BASE/webapps/ROOT/WEB-INF/apsv-*.xml`
  - Remote administration

PROGRESS EXCHANGE 2014

Visit the Resource Portal

- Get **session details** & presentation **downloads**
- Complete a **survey**
- Access the latest Progress **product literature**

www.progress.com/exchange2014

PROGRESS