PUG CHALLENGE EXCHANGE AMERICAS

Moving On Up to PASOE/Web

Migrating WebSpeed Applications to Progress Application Server

Overview

Progress Application Server Platform

- A single delivery platform for all Progress Web-based products
- Not only the application but also the web server to support it
  - Created from Apache Tomcat 7.0.55 distribution
- Designed for secure operation
  - Spring Security Framework included
  - Realms and roles defined to implement access control

High-level features

- Secure
- Simple
  - 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

PASOE/Web

The return of WebSpeed

(hooray)

Why hooray?

All the methods (verbs)

GET
PUT
POST
DELETE
HEAD
PATCH
OPTIONS
TRACE

Why hooray?

All the methods (verbs)

All the message

GET
PUT
POST
DELETE
HEAD
PATCH
OPTIONS
TRACE

```
POST /rest/EmployeeSvc/Employee HTTP/1.1
Authorization: Basic dG9tY2F0OnRvbWNhdA==
Host: oelxdev06:8881
User-Agent: OpenEdge-HttpClient/0.3.0
Content-Length: 18

{"request": "data"}
```

Why hooray?

All the methods (verbs)

All the message

All the control

GET HEAD
PUT PATCH

```
define variable webRequest as IWebReq
define variable httpStatus as integer
assign webRequest = new WebRequest()
       httpStatus = integer(StatusCod
case MethodEnum:GetEnum(webRequest:Me
     when MethodEnum:DELETE  then htt
     when MethodEnum:GET     then htt
     when MethodEnum:HEAD    then htt
     when MethodEnum:POST    then htt
     when MethodEnum:PUT     then htt
     otherwise                    htt
end case.
Assert:NotNull(httpStatus, 'Status re
/* good */
return httpStatus.
catch oError as Progress.Lang.Error:
    assign httpStatus = HandleExcepti
    if httpStatus eq ? then
        assign httpStatus = 0.
    return httpStatus.
end catch.
```

```
oHeader = poRequest:GetHeader('Content-Type').
oHeader:ParamDelimiter = ';':u.
oHeader:ExtractParameters().

/* URL is /web/img/Employee/{EmpNum} */
assign iEmpNum = integer(poRequest:GetPathParameter('EmpNum':u))
       oEntityWriter = EntityWriterBuilder:Build(poRequest)
                        :Option('multipartBoundary':u,
oHeader:GetParameterValue('boundary':u))
                        :Writer.
oEntityWriter:Open().
oEntityWriter:Write(poRequest:Entity).
oEntityWriter:Close().

assign oEntity = cast(oEntityWriter:Entity, MultipartEntity)
       oPart   = oEntity:GetPart(1)
       oHeader = oPart:Headers:Get('Content-Disposition':u)

       /* Content-Disposition: form-data; name="myphoto.png";
filename="emp_21.png" */
       cImageFileName = oHeader:GetParameterValue('filename':u).

moBE:WriteEmployeePic(iEmpNum,
                      cImageFileName,
                      cast(oPart:Body, ByteBucket):GetBytes()).
```

But Wait! There's More!

All the meth

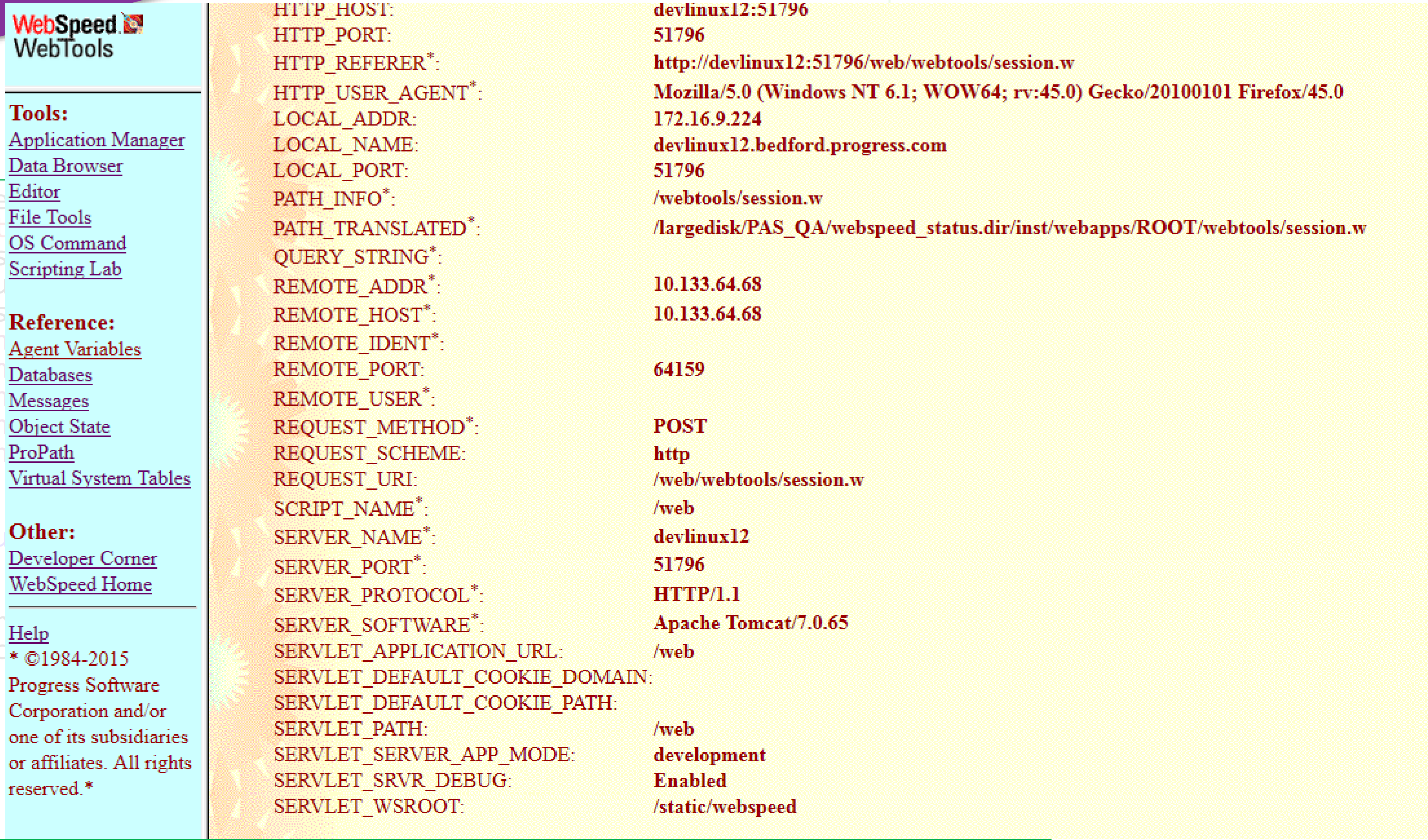
ge

ll the control

HEAD
PUT
PATCH

```
define variable webRequest as IWe
define variable httpStatus as int
assign webRequest = new WebReques
       httpStatus = integer(Statu
case MethodEnum:GetEnum(webReques
     when MethodEnum:DELETE  then
     when MethodEnum:GET     then
     when MethodEnum:HEAD    then
     when MethodEnum:POST    then
     when MethodEnum:PUT     then
     otherwise
end case.
Assert:NotNull(httpStatus, 'Statu
/* good */
return httpStatus.
catch oError as Progress.Lang.Err
    assign httpStatus = HandleExc
    if httpStatus eq ? then
        assign httpStatus = 0.
    return httpStatus.
end catch.
```

Your existing WebSpeed just works!



WebSpeed WebTools

Tools:
Application Manager
Data Browser
Editor
File Tools
OS Command
Scripting Lab

Reference:
Agent Variables
Databases
Messages
Object State
ProPath
Virtual System Tables

Other:
Developer Corner
WebSpeed Home

Help
* ©1984-2015 Progress Software Corporation and/or one of its subsidiaries or affiliates. All rights reserved.*

HTTP_HOST:	devlinux12:51796
HTTP_PORT:	51796
HTTP_REFERER*:	http://devlinux12:51796/web/webtools/session.w
HTTP_USER_AGENT*:	Mozilla/5.0 (Windows NT 6.1; WOW64; rv:45.0) Gecko/20100101 Firefox/45.0
LOCAL_ADDR:	172.16.9.224
LOCAL_NAME:	devlinux12.bedford.progress.com
LOCAL_PORT:	51796
PATH_INFO*:	/webtools/session.w
PATH_TRANSLATED*:	/largedisk/PAS_QA/webspeed_status.dir/inst/webapps/ROOT/webtools/session.w
QUERY_STRING*:	
REMOTE_ADDR*:	10.133.64.68
REMOTE_HOST*:	10.133.64.68
REMOTE_IDENT*:	
REMOTE_PORT:	64159
REMOTE_USER*:	
REQUEST_METHOD*:	POST
REQUEST_SCHEME:	http
REQUEST_URI:	/web/webtools/session.w
SCRIPT_NAME*:	/web
SERVER_NAME*:	devlinux12
SERVER_PORT*:	51796
SERVER_PROTOCOL*:	HTTP/1.1
SERVER_SOFTWARE*:	Apache Tomcat/7.0.65
SERVLET_APPLICATION_URL:	/web
SERVLET_DEFAULT_COOKIE_DOMAIN:	
SERVLET_DEFAULT_COOKIE_PATH:	
SERVLET_PATH:	/web
SERVLET_SERVER_APP_MODE:	development
SERVLET_SRVR_DEBUG:	Enabled
SERVLET_WSROOT:	/static/webspeed

Moving parts

Request flow … incoming

1. Instance receives the request

```
POST /CorpHR/web/Employees/pjudge HTTP/1.1
```

2. ABL application config maps the URI to an ABL handler

```
handler1=Sports.Web.EmployeeHandler: /Employees/{EmpName}
```

3. Handler calls a HTTP method-appropriate method

```
method override protected integer HandlePost(
                              poRequest as IWebRequest)
```

4. HandlePost() reads the request and calls appropriate business logic

```
oHeader = poRequest:GetHeader('Content-Type').
moBE:WriteEmployeePic(iEmpNum, cImageFileName, oPic).
moBE:GetEmployeeInfo(iEmpNum, output oEmployeeJsonData).
```

Request flow … incoming

Concepts

- Web handler

1. Instance receives the request

   ```
   POST /CorpHR/web/Employees/pjudge HTTP/1.1
   ```

2. ABL application config maps the URI to an ABL handler

   ```
   handler1=Sports.Web.EmployeeHandler: /Employees/{EmpName}
   ```

3. Handler calls a HTTP method-appropriate method

   ```
   method override protected integer HandlePost(
                                  poRequest as IWebRequest)
   ```

4. HandlePost() reads the request and calls appropriate business logic

   ```
   oHeader = poRequest:GetHeader('Content-Type').
   moBE:WriteEmployeePic(iEmpNum, cImageFileName, oPic).
   moBE:GetEmployeeInfo(iEmpNum, output oEmployeeJsonData).
   ```

Request flow … incoming

1. Instance receives the request

```
POST /CorpHR/web/Employees/pjudge HTTP/1.1
```

2. ABL application config maps the URI to an ABL handler

```
handler1=Sports.Web.EmployeeHandler: /Employees/{EmpName}
```

3. Handler calls a HTTP method-appropriate method

```
method override protected integer HandlePost(
                              poRequest as IWebRequest)
```

4. HandlePost() reads the request and calls appropriate business logic

```
oHeader = poRequest:GetHeader('Content-Type').
moBE:WriteEmployeePic(iEmpNum, cImageFileName, oPic).
moBE:GetEmployeeInfo(iEmpNum, output oEmployeeJsonData).
```

Concepts

- Web handler
- Handle* method

Request flow … incoming

1. Instance receives the request

```
POST /CorpHR/web/Employees/pjudge HTTP/1.1
```

2. ABL application config maps the URI to an ABL handler

```
handler1=Sports.Web.EmployeeHandler: /Employees/{EmpName}
```

3. Handler calls a HTTP method-appropriate method

```
method override protected integer HandlePost(
                                    poRequest as IWebRequest)
```

4. HandlePost() reads the request and calls appropriate business logic

```
oHeader = poRequest:GetHeader('Content-Type').
moBE:WriteEmployeePic(iEmpNum, cImageFileName, oPic).
moBE:GetEmployeeInfo(iEmpNum, output oEmployeeJsonData).
```

Concepts

- Web handler
- Handle* method
- IWebRequest

Request flow … outgoing

5. HandlePost() creates a response object

```
assign oResp                = new OpenEdge.Web.WebResponse()
       oResp:Entity         = oEmployeeJsonData
       oResp:ContentType = 'application/json':u.
```

6. HandlePost() writes the response data to the output stream

```
oWriter = new WebResponseWriter(oResp).
oWriter:Open().
cast(oResp:Entity, JsonObject):Write(mEntity).
oResp:ContentLength = get-size(mEntity).
oWriter:Write(mEntity).
return integer(StatusCodeEnum:None). // 0 - zero
```

7. Alternative approach for errors/status codes

```
return integer(StatusCodeEnum:InternalServerError). // 500
```

Concepts

- Web handler
- Handle* method
- IWebRequest

Request flow … outgoing

5. HandlePost() creates a response object

```
assign oResp             = new OpenEdge.Web.WebResponse()
       oResp:Entity      = oEmployeeJsonData
       oResp:ContentType = 'application/json':u.
```

6. HandlePost() writes the response data to the output stream

```
oWriter = new WebResponseWriter(oResp).
oWriter:Open().
cast(oResp:Entity, JsonObject):Write(mEntity).
oResp:ContentLength = get-size(mEntity).
oWriter:Write(mEntity).
return integer(StatusCodeEnum:None). // 0 - zero
```

7. Alternative approach for errors/status codes

```
return integer(StatusCodeEnum:InternalServerError). // 500
```

Concepts

- Web handler
- Handle* method
- IWebRequest
- Web Response object

Request flow … outgoing

5. HandlePost() creates a response object

```
assign oResp              = new OpenEdge.Web.WebResponse()
       oResp:Entity       = oEmployeeJsonData
       oResp:ContentType = 'application/json':u.
```

6. HandlePost() writes the response data to the output stream

```
oWriter = new WebResponseWriter(oResp).
oWriter:Open().
cast(oResp:Entity, JsonObject):Write(mEntity).
oResp:ContentLength = get-size(mEntity).
oWriter:Write(mEntity).
return integer(StatusCodeEnum:None). // 0 - zero
```

7. Alternative approach for errors/status codes

```
return integer(StatusCodeEnum:InternalServerError). // 500
```

Concepts

- Web handler
- Handle* method
- IWebRequest
- Web Response object
- Web Response Writer

Request flow … outgoing

5. HandlePost() creates a response object

```
assign oResp             = new OpenEdge.Web.WebResponse()
       oResp:Entity      = oEmployeeJsonData
       oResp:ContentType = 'application/json':u.
```

6. HandlePost() writes the response data to the output stream

```
oWriter = new WebResponseWriter(oResp).
oWriter:Open().
cast(oResp:Entity, JsonObject):Write(mEntity).
oResp:ContentLength = get-size(mEntity).
oWriter:Write(mEntity).
return integer(StatusCodeEnum:None). // 0 - zero
```

7. Alternative approach for errors/status codes

```
return integer(StatusCodeEnum:InternalServerError). // 500
```

Concepts

- Web handler
- Handle* method
- IWebRequest
- Web Response object
- Web Response Writer
- Status Codes

What is a web handler?

- Performs request routing & parameter mapping
  - Figures out what to run
  - Converts HTTP into ABL
- OOABL implementation of Progress.Web.IWebRequest ← for advanced use only
- In-the-box versions include
  - OpenEdge.Web.WebHandler ← use this for new code
    - Abstract class with some default behaviour
  - OpenEdge.Web.CompatibilityHandler ← what 'classic' WebSpeed uses
  - OpenEdge.Web.DefaultHandler ← locked-out version

- OpenEdge.Web & OpenEdge.Net packages in $DLC/tty/netlib/OpenEdge.Net.pl
- API doc at https://documentation.progress.com/output/oehttpclient/

Which handler is used?

✓ Web handler

```
defaultHandler=OpenEdge.Web.DefaultHandler
handler1=Sports.Web.SportsHandler: /{resources}/catalog/{service}
handler2=Sports.Web.SportsHandler: /Customers/catalog/updateSvc
```

- handler*N*=ooabl.type.name : <relative-uri>
- <relative-uri>
  - Relative to **`/web`** ← the transport path
  - Needs leading /
  - Text "customer"
  - Tokens supported: {CustomerName} or {pathparam: regex}
- Matched in (numeric) order and then by best match
  - Handlers may be reused for differing paths
- Each webapp has a default for no-match-found

Where does my WebHandler live?

Web handler

- We consider it a Service Interface
  - So in the webapp - WEB-INF/openedge

- But it's Just ABL so can be anywhere on PROPATH
  - Packages
  - Propath

Coding a web handler

✓ Handle* method

1. Create a new class that inherits from OpenEdge.Web.WebHandler
2. You must implement 3 methods …
   1. HandleGet
   2. HandleMethodNotImplemented
   3. HandleNotAllowed
3. … but you'll want to implement more
   - Handle<http-method>
   - HandleException

Coding a web handler (generated code)

Handle* method

```
METHOD OVERRIDE PROTECTED INT HandleGet(INPUT poRequest AS OpenEdge.Web.IWebRequest ):
  DEFINE VARIABLE oResponse AS OpenEdge.Net.HTTP.IHttpResponse NO-UNDO.
  DEFINE VARIABLE oWriter   AS OpenEdge.Web.WebResponseWriter  NO-UNDO.
  DEFINE VARIABLE oBody     AS OpenEdge.Core.String            NO-UNDO.
  /* The WebResponse body is a wrapper around an entire HTTP response message. It contains a status code and
reason; headers; cookies and a message body. API-level doc for this and related classes can be found at
https://documentation.progress.com/output/oehttpclient/ */
  ASSIGN oResponse            = NEW OpenEdge.Web.WebResponse()
         oResponse:StatusCode = INTEGER(StatusCodeEnum:OK)
         /* This body object can be a string or something else (JsonObject for instance) */
         oBody = NEW OpenEdge.Core.String('Hello Administrator')
         oResponse:Entity        = oBody
         oResponse:ContentType   = 'text/plain':u /* HTTP messages require a content type */
         oResponse:ContentLength = oBody:Size /* ContentLength is good too */
         /* The WebResponseWriter ensures that the status line and all headers are written out before the
            message body/entity. */
        ASSIGN oWriter = NEW WebResponseWriter(oResponse).
        oWriter:Open().
        /* The Progress.IO.OutputStream Write() methods take multiple overloads, for
           a variety of data types. See the doc for more information. */
        oWriter:Write(oBody:Value).
        /* Finish writing the response message */
        oWriter:Close().
        /* A response of 0 means that this handler will build the entire response;
           a non-zero value is mapped to a static handler in the webapp's /static/error folder.
           The mappings are maintained in the webapps's WEB-INF/web.xml
           A predefined set of HTTP status codes is provided in the OpenEdge.Net.HTTP.StatusCodeEnum
           enumeration */
        RETURN 0.

 END METHOD.
```

Coding a web handler (cont'd)

Handle* method

```
METHOD OVERRIDE PROTECTED INTEGER HandleNotImplemented(
                              INPUT poRequest AS OpenEdge.Web.IWebRequest ):

/* Throwing an error from this method results in a 500/Internal Server Error response. The
web handler will attempt to log this exception. See the HandleGet method's comments on
choosing a value to return from this method. */

  UNDO, THROW NEW Progress.Lang.AppError("METHOD NOT IMPLEMENTED").

END METHOD.
```

Incoming data – OpenEdge.Web.IWebRequest

IWebRequest object

Message element	oRequest = new WebRequest()	
HTTP method ("verb")	:Method	"POST"
URL	:URI	http://localhost:8810/SportsSvc/web/Customer/catalog?filter={"ablWhere"...}
Query parameters	:URI:GetQueryNames() :URI:GetQueryValue	["filter"] Filter => " {'ablWhere':'custnum eq 42"} "
Headers	:GetHeaders() :GetHeader(<name>):Value	[HttpHeader, HttpHeader, HttpHeader] Accept => "application/json"
Cookies	:GetCookie(<name>)	
Path parameters	PathParameterNames GetPathParameter(<name>)	"Resources,service" "Customer"
Entity / message body	ContentType / ContentLength Entity	application/json
Path information	TransportPath PathInfo WebAppPath	/web /Customer/catalog /SportsSvc

Outgoing data – IWebResponse

✓ Web Response object

- If a request asks the question "please do something for me", a response is the answer
- You have TOTAL control over …
  - Status codes: 200/OK, 201/Created, 418/I'm a teapot, 501/Not implemented
  - Headers & cookies
  - Entities (payload/body) – can be anything (almost)
    - HTML, JSON, text, XML, multipart, binary, …
  - Response chunking

Return to sender

✓ Web Response Writer

- OpenEdge.Web.WebResponseWriter a built-in class
- Writes the HTTP 'preamble' nicely
  - Status line `HTTP/1.1 200 OK`
  - Headers
  - Cookies
- Choose how the body is written

Single Write(<data>) call	Multiple Write(<data>) calls
Gather entire body before sending	Enables HTTP chunking
Choose your error-page strategy	Cannot use PASOE static error pages
Set all the meta-data before you need writing	Preamble written on first subsequent Write()

- Very similar in many senses to `{&OUT}` approach for 'cgi-wrapper' WebSpeed

Returning errors

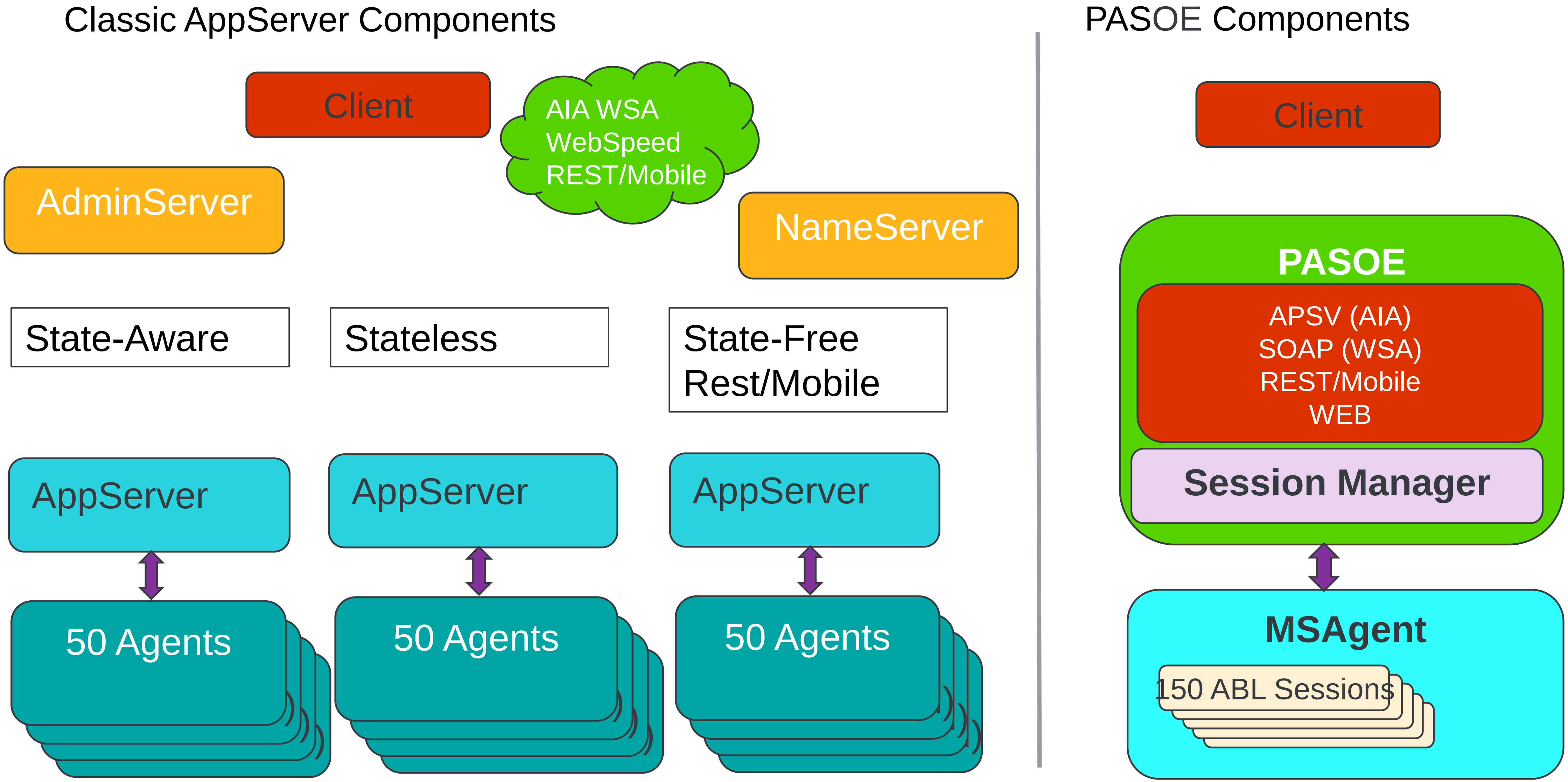
- An error can be Just Another Response
  - Set the StatusCode to 5xx or 4xx and add an entity (or not)
  - RETURN 0
  - Not always suited for standardized responses

- You can return a static error page
  - Handler returns INTEGER value
  - Set up in web.xml ← per webapp in WEB-INF/
  - Static page is returned ← by default in WEB-INF/jsp/

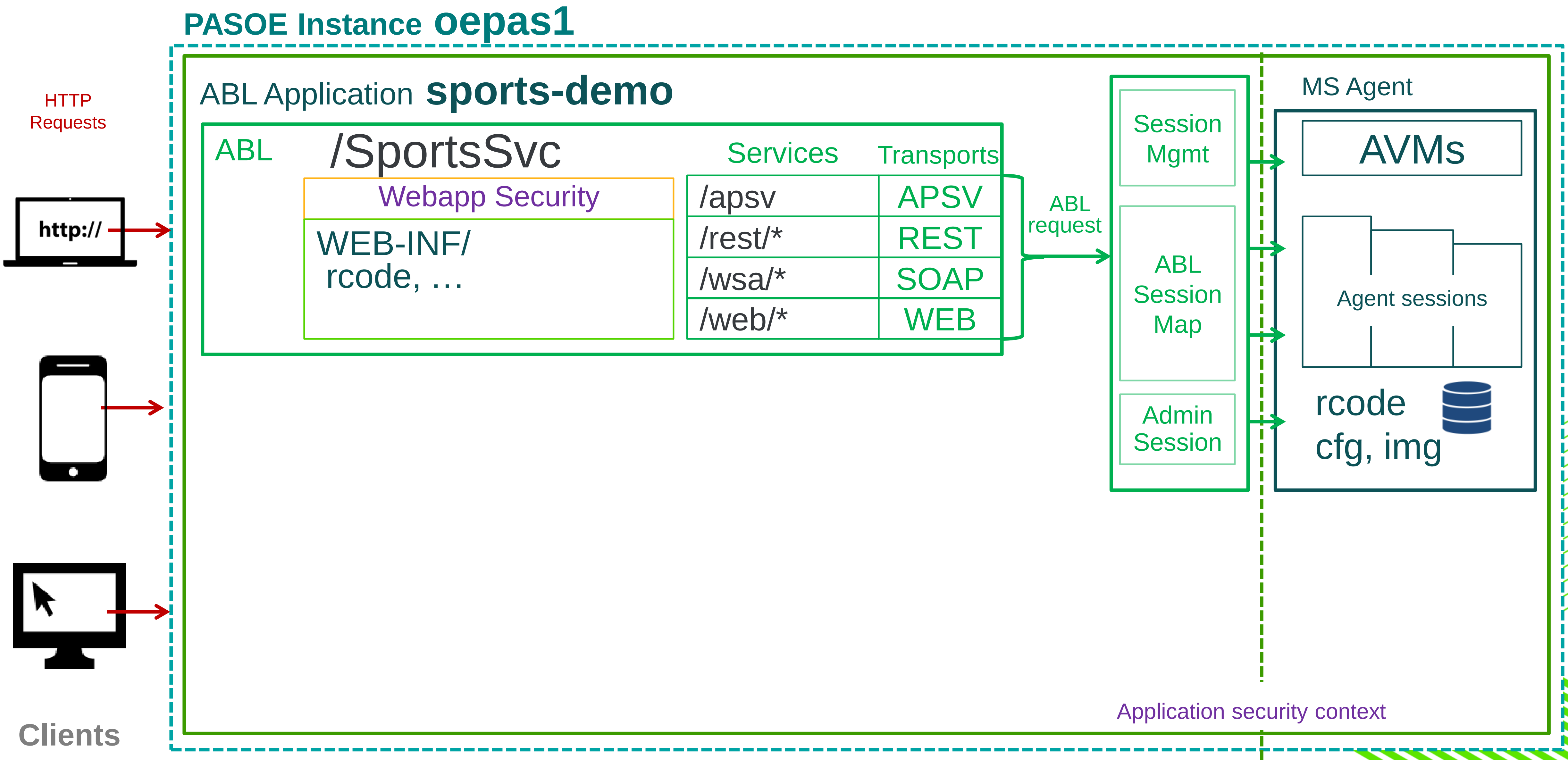
```
<error-page>
   <location>/WEB-INF/jsp/errorPage.jsp</location>
 </error-page>
```

Migration

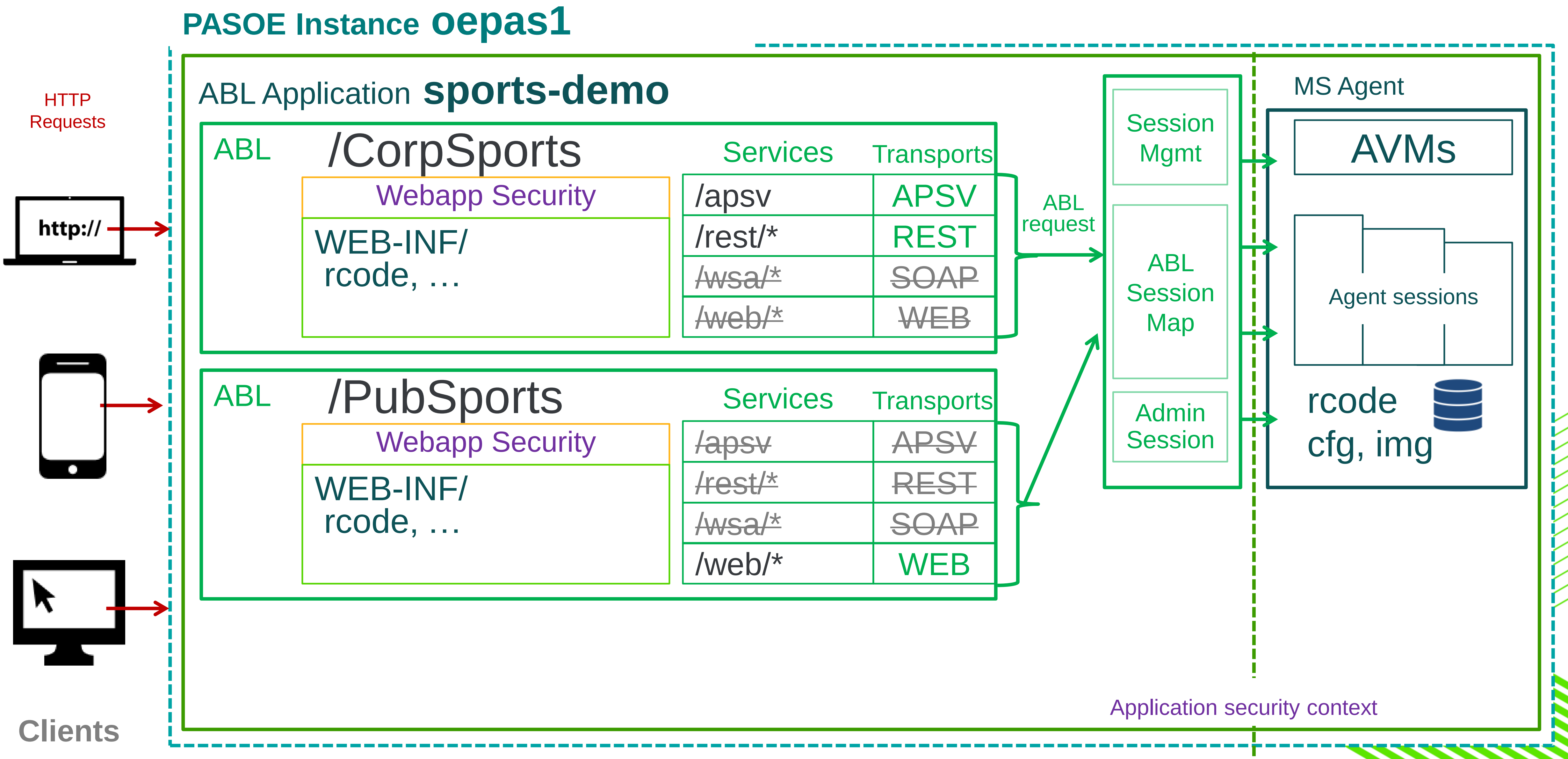
Application server components



Instances, web-apps and ABL applications



Instances, web-apps and ABL applications



PASOE Instance **oepas1**

HTTP Requests

Clients

ABL Application **sports-demo**

ABL /CorpSports

Webapp Security

WEB-INF/
rcode, …

Services	Transports
/apsv	APSV
/rest/*	REST
~~/wsa/*~~	~~SOAP~~
~~/web/*~~	~~WEB~~

ABL /PubSports

Webapp Security

WEB-INF/
rcode, …

Services	Transports
~~/apsv~~	~~APSV~~
~~/rest/*~~	~~REST~~
~~/wsa/*~~	~~SOAP~~
/web/*	WEB

ABL request

Session Mgmt

ABL Session Map

Admin Session

MS Agent

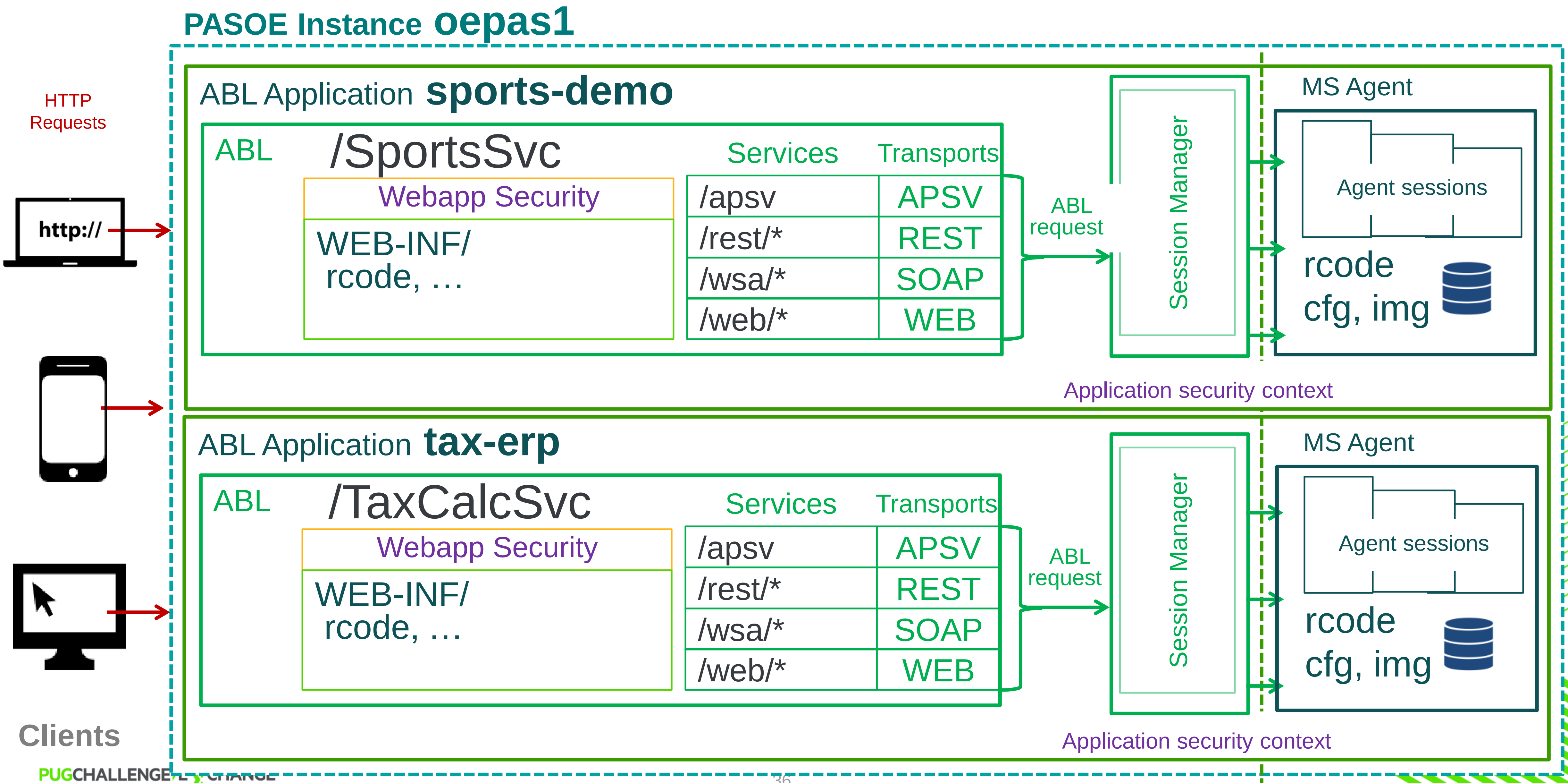
AVMs

Agent sessions

rcode
cfg, img

Application security context

Instances, web-apps and ABL applications



PASOE Instance oepas1

HTTP Requests

http://

Clients

ABL Application **sports-demo**

ABL /SportsSvc

Webapp Security

WEB-INF/
rcode, …

Services	Transports
/apsv	APSV
/rest/*	REST
/wsa/*	SOAP
/web/*	WEB

ABL request

Session Manager

MS Agent

Agent sessions

rcode
cfg, img

Application security context

ABL Application **tax-erp**

ABL /TaxCalcSvc

Webapp Security

WEB-INF/
rcode, …

Services	Transports
/apsv	APSV
/rest/*	REST
/wsa/*	SOAP
/web/*	WEB

ABL request

Session Manager

MS Agent

Agent sessions

rcode
cfg, img

Application security context

Migration: architecture

- Classic
  - One call from www -> cgi -> web-disp.p
  - Process, session, request all same thing
  - Wait-for web-notify
- PASOE/WEB
  - Agent and Session and request totally separate
  - Sessions/processes persist beyond request
  - Classic request/response
  - SESSION:REMOTE, ~CLIENT-TYPE, ~SERVER-OPERATING-MODE checks
  - SESSION:CURRENT-REQUEST-INFO:AdapterType

Migration: Session state

- Classic
  - Can only be stateless

- PASOE
  - Has to be session-free
  - Means requests aren't guaranteed to go to the same session
  - WebSpeed global variables may need (re)setting in different places
  - Beware of cleanup code being excessively enthusiastic

PAS for OpenEdge Production versus Development Products

PAS for OE Development	PAS for OE Production
Can compile code	Cannot compile code
Non-secure configuration	Secure configuration
Test server instance in $WRKDIR	No test server instances
Remote administration included Tomcat remote admin enabled OpenEdge remote admin enabled	Remote administration optional Tomcat remote admin optional OpenEdge remote admin optional
Built-in oeabl web application (ROOT) All transport deployed and enabled	Built-in oeabl web application (ROOT) All transports deployed but disabled
Restricted 5 concurrent requests 1 agent	Unrestricted concurrent requests number of agents

Summary

- WEB transport gives you the capabilities to handle requests from any client
  - Complete control over the 'gazinters' and 'gazouters'

- In parallel to your APSV clients
  - With the same or different security models

- PASOE gives you the ability to tailor the deployment of your ABL applications
  - Depending on client sizes
  - Depending on you scalability needs
  - Depending on the developer-deployer-model

Questions?

PUGCHALLENGE AMERICAS EXCHANGE

Peter Judge pjudge@progress.com

Security

Spring Security

- OpenEdge supplements the Java container’s security with the industry-recognized Spring security framework
- Spring Security is a customizable authentication and access control framework

  http://projects.spring.io/spring-security/
- It is one of the industry standards for securing Spring-based applications

Web Application Authentication Models

- Anonymous : No user authentication or login session
- HTTP **Basic** Authentication — Client sends base64 encoded user name/password to web application in each http request
  - HTTP header: Authorization
- HTTP **Form** Authentication — Client logs in and out the web application once per session
  - Login: The client obtains user credentials and POSTs them to the web application
    - URI: /static/auth/j_spring_security_check
    - Body: j_username=xxxx&j_password=yyyy&submit=Submit+Query
    - Cookie: JSESSIONID
  - Logout: The client uses a GET request to log out
    - URI: /static/auth/j_spring_security_logout