Progress®

OERA, PASOE & (some) good (best?) practices

Architecture, structure, monitoring, security

Peter Judge, Progress Software
pjudge@progress.com

$ whoami

pjudge@progress.com

Software Architect*

@ Progress since 2003

Integration-y stuff – Authentication Gateway, HTTP-Out, Corticon et al

OE Best Practices / OERA / AutoEdge / CCS

4GL since 1996

* Aka programmer who knows PowerPoint

Agenda

- OpenEdge Reference Architecture & its evolution
- Application structure & design
- Monitoring
- Troubleshooting & Logging
- Security

What is the OERA?

The OpenEdge Reference Architecture (OERA) defines the general **functional categories** of components that comprise an application. It can be used as a high-level **blueprint** for developing OpenEdge **service-oriented** business applications.

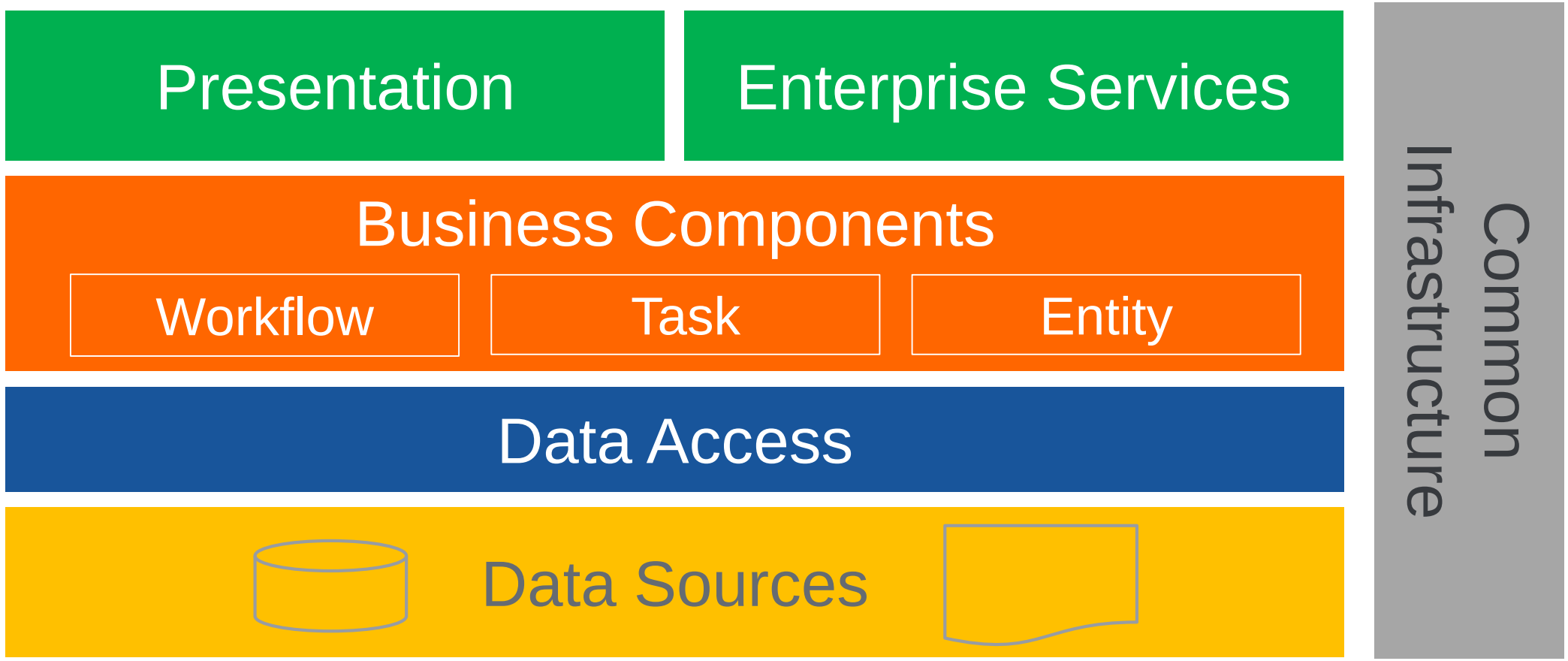
Each layer of the OERA consists of distinct components, each with specific characteristics, roles and responsibilities. In addition, the OERA provides guidelines as to how each of the architectural components interacts. The following diagram illustrates the component architecture and the relationships between each of the components.

https://community.progress.com/community_groups/openedge_architecture/w/openedgecloudarcade/866.introduction-to-the-openedge-reference-architecture

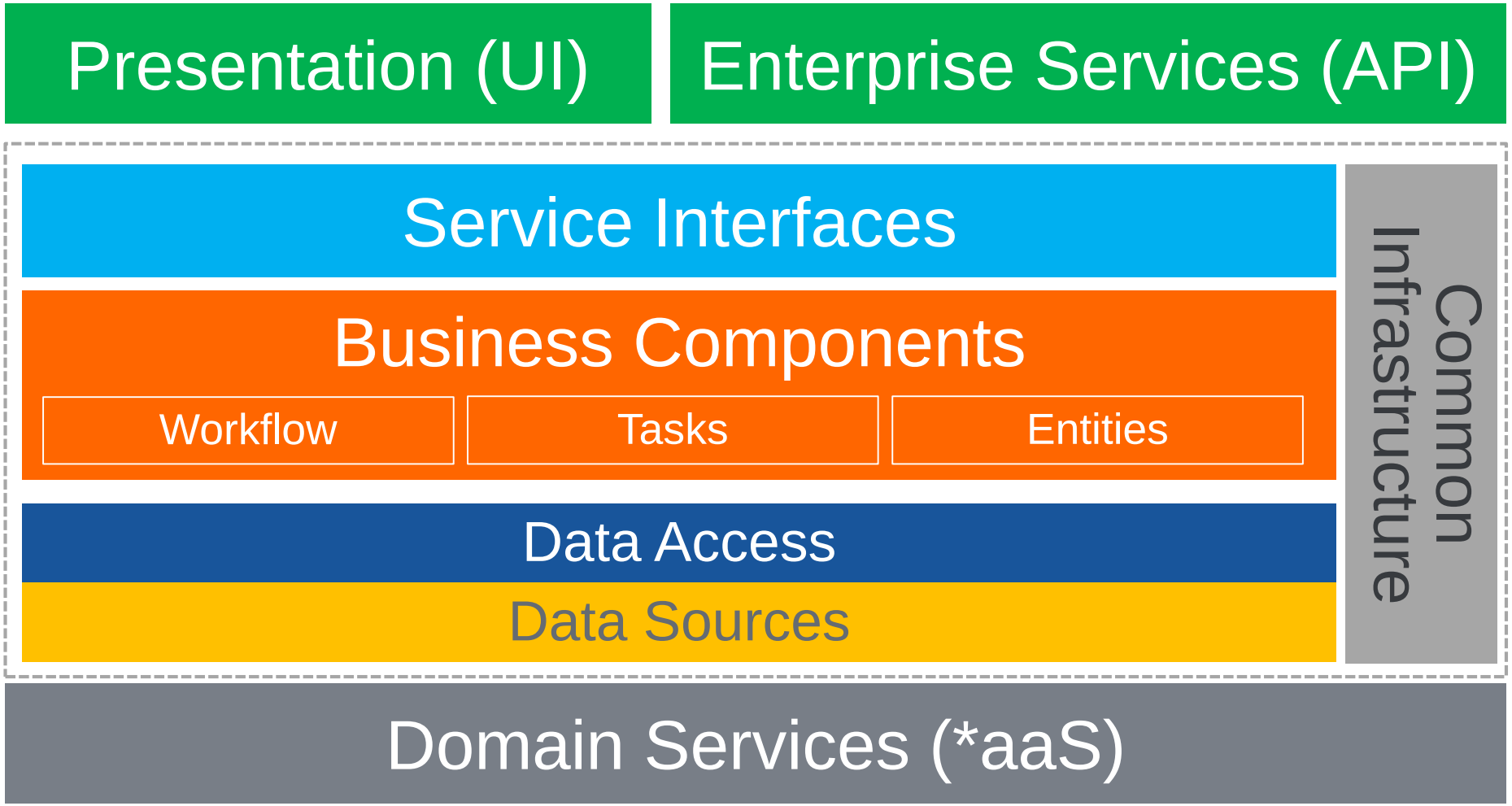
A Brief History of OERA

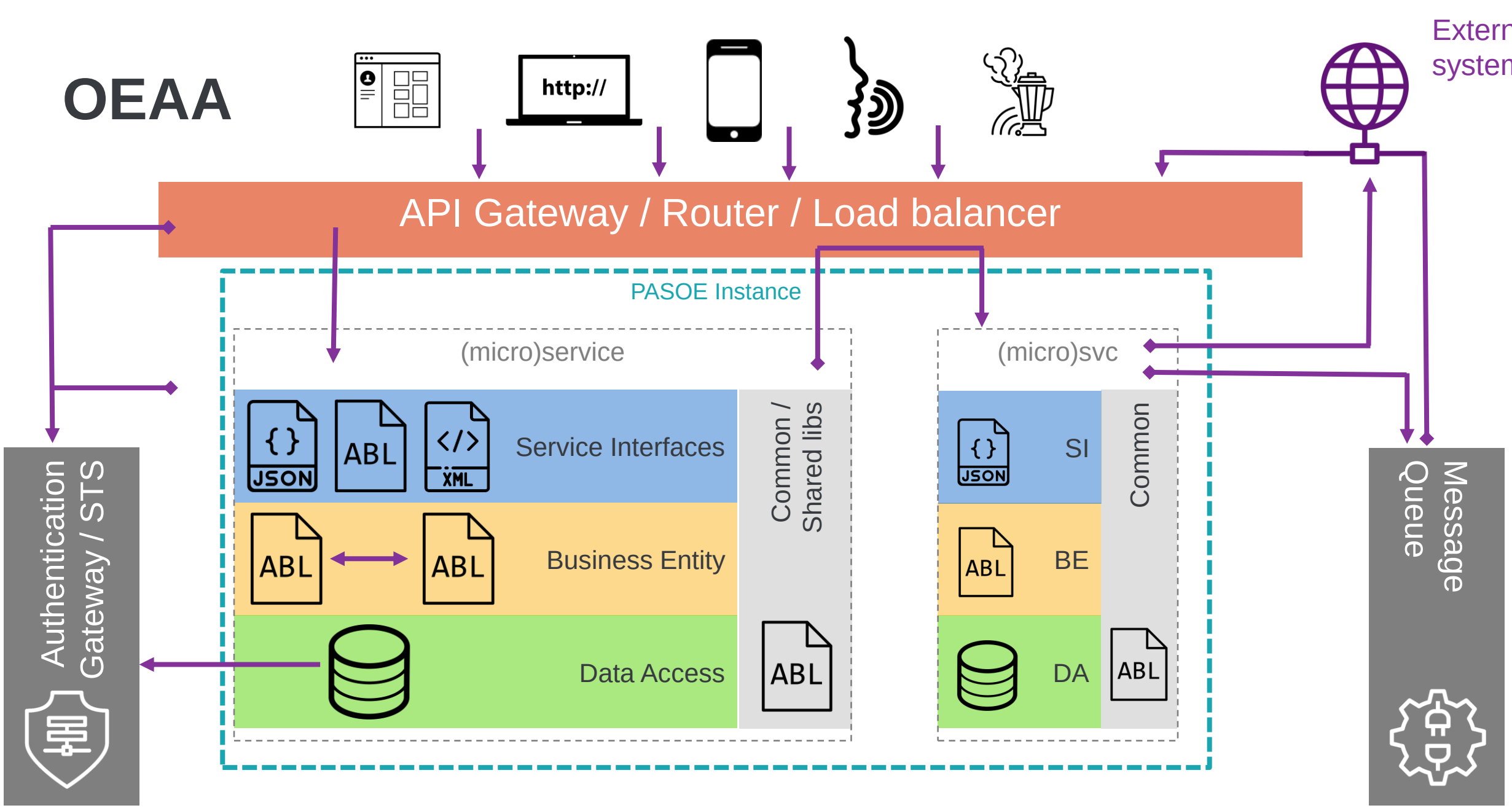
- White-Papers initially published by John Sadd and Mike Omerod
- Training material, not a product component
  - Guidance for building it yourself
- Limited support in the products
  - Tools for Business Logic (T4BL) in PDSOE
  - DataView SmartObject
- “AutoEdge” sample implementation
  - Dealer & Factory applications
- https://community.progress.com/community_groups/openedge_architecture/

OpenEdge Reference Architecture (OERA)



OpenEdge Reference Architecture (OERA)

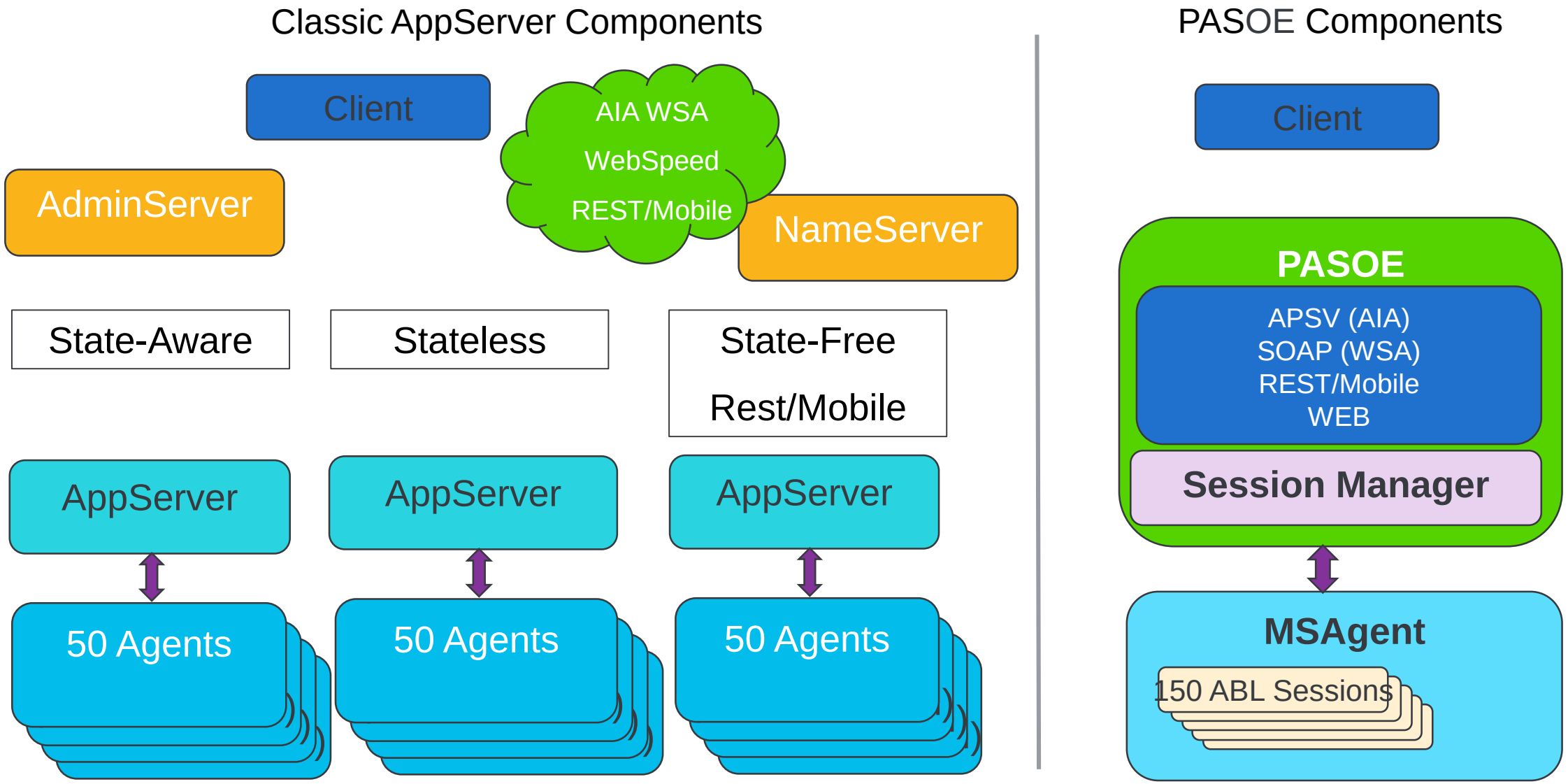




Progress®

Progress Application Server for OpenEdge

Application server components



Application and URI design

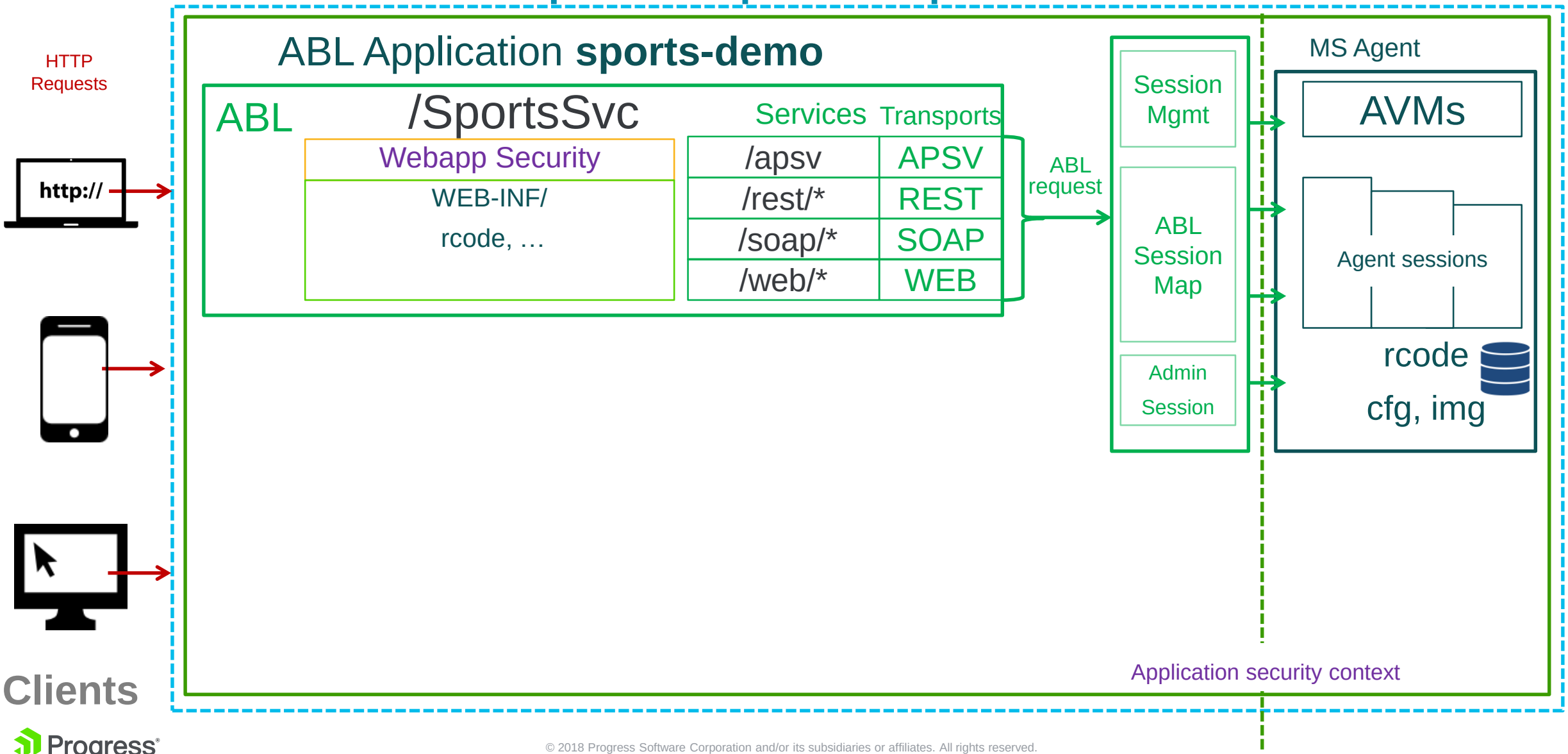
- Webapps and URI spaces
- Number of ABL applications
- Deployment aspects

Administrator

Application Developer

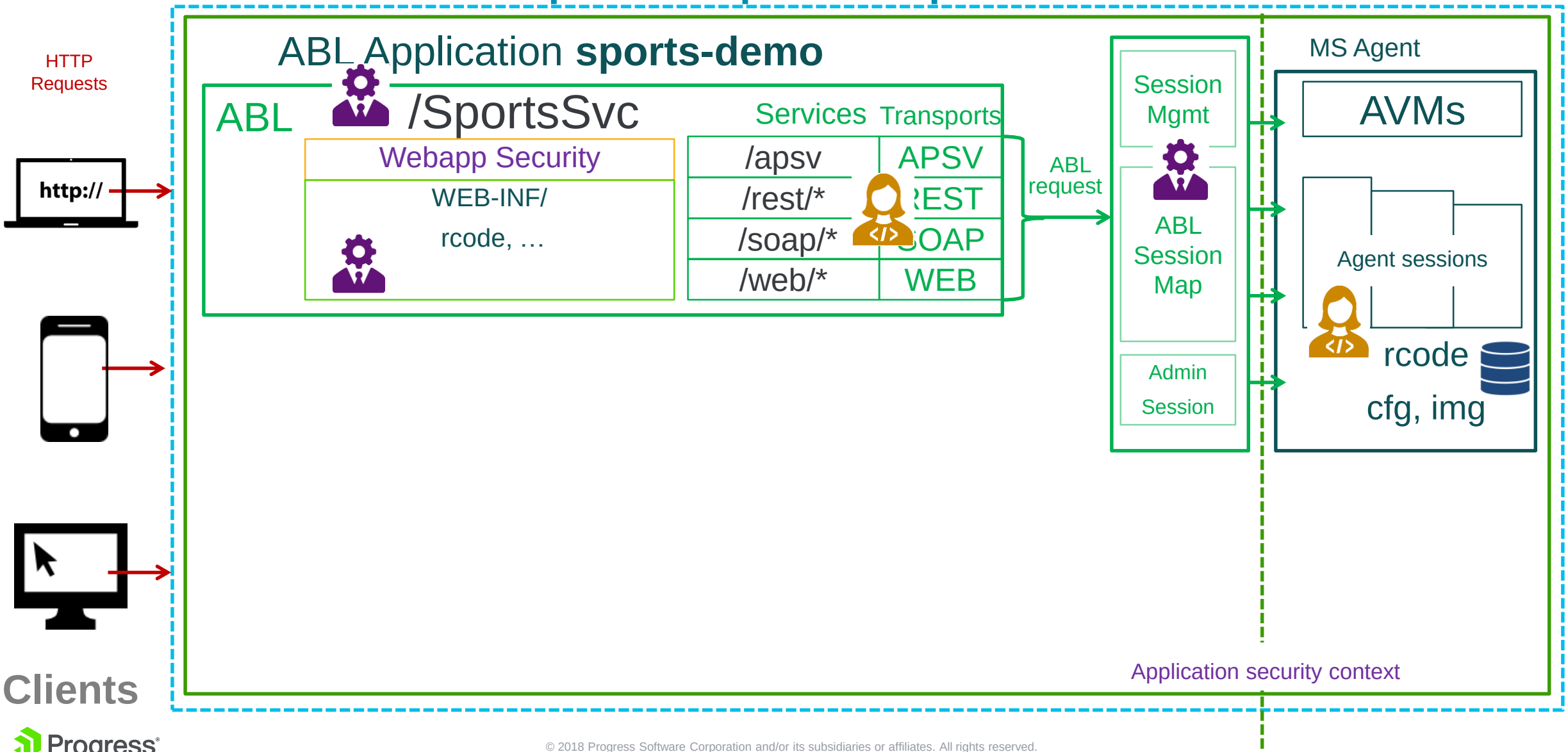
Instances, web-apps and ABL applications

PASOE Instance **oepas1 / https://example.com:8810**

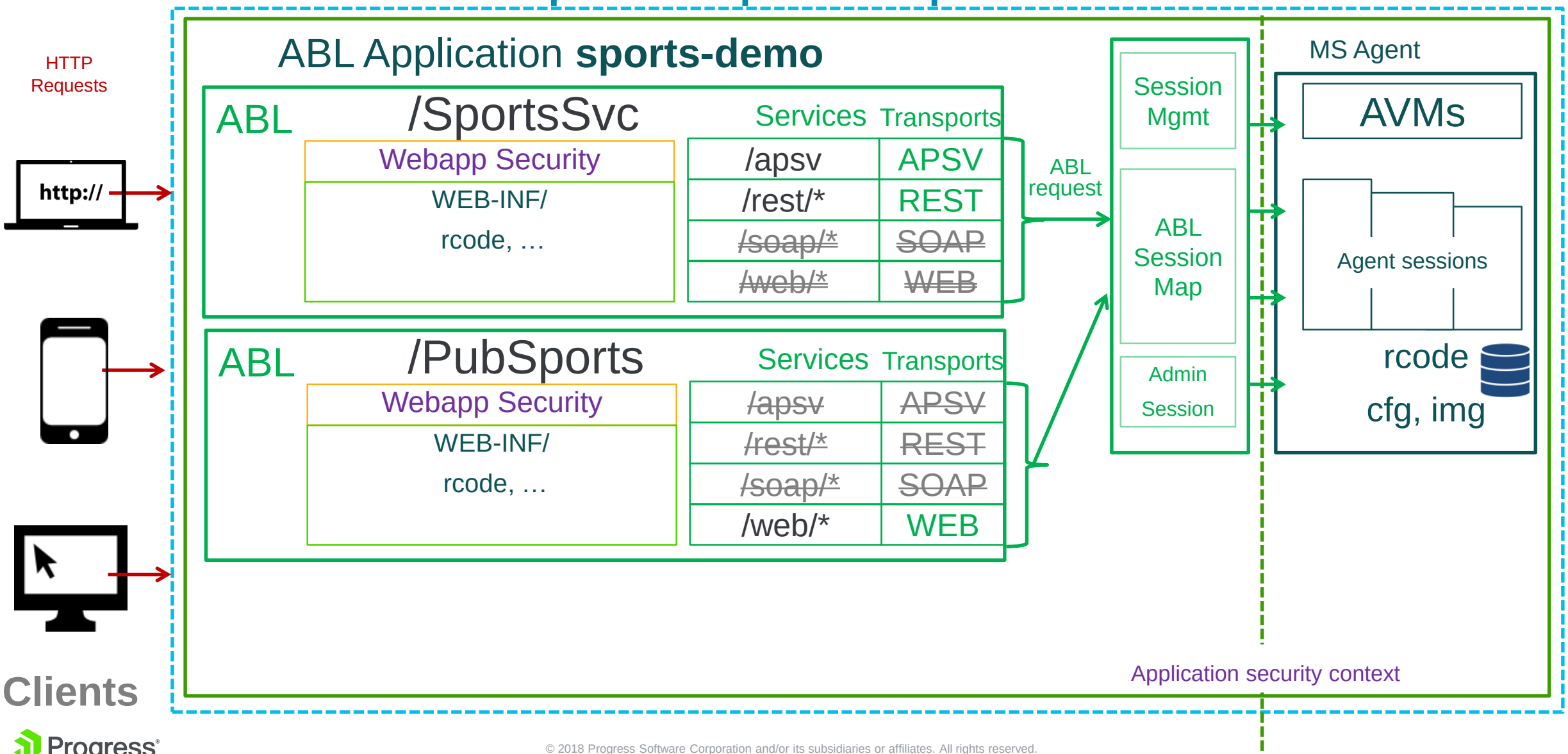


Instances, web-apps and ABL applications

PASOE Instance **oepas1 / https://example.com:8810**

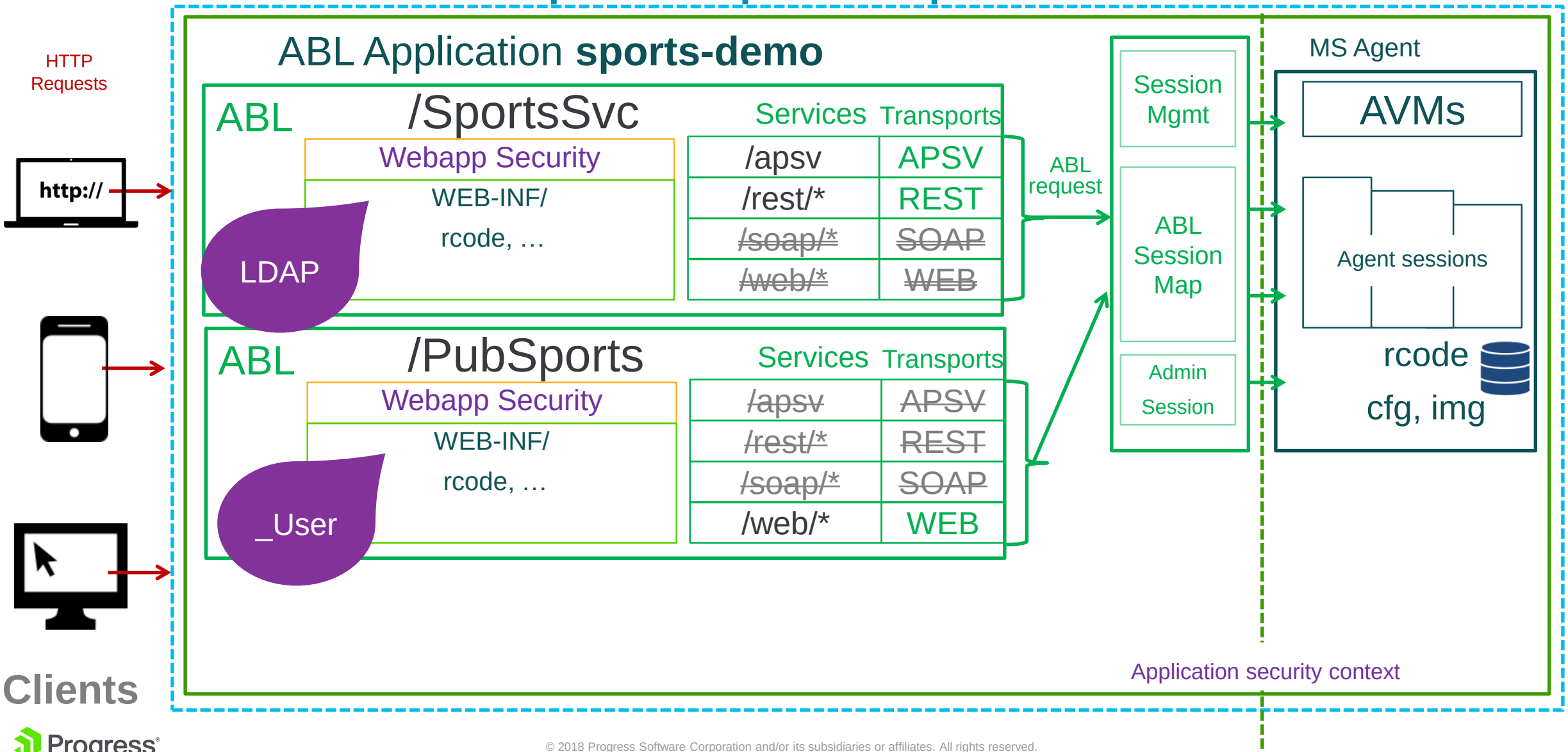


Instances, web-apps and ABL applications



Instances, web-apps and ABL applications

PASOE Instance **oepas1 / https://example.com:8810**



Service URLs

http://example.com:8810/SportsSvc/rest/CustomerBE/Customer

PASOE-instance

ABL-webapp

Transport

ABL-service

sports-demo

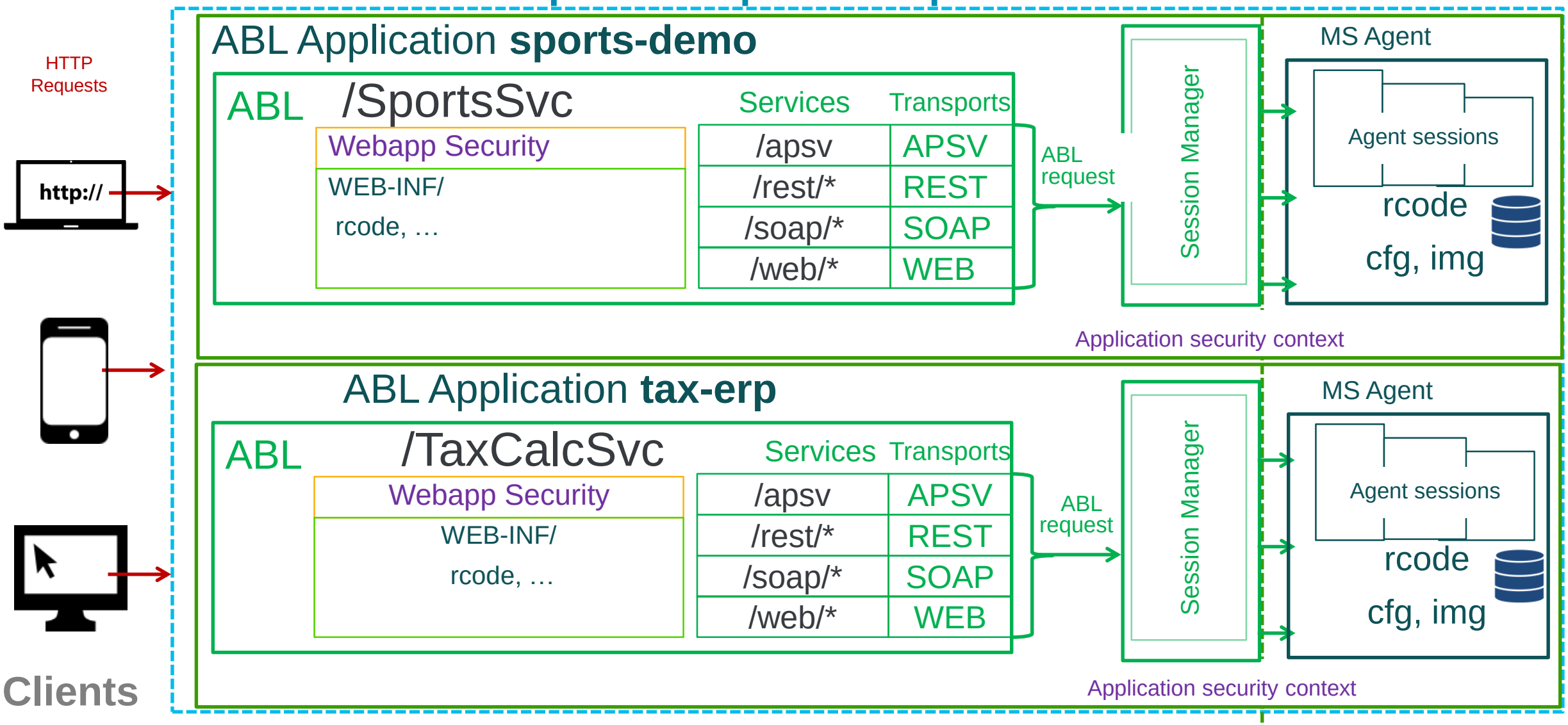
ABL-application

- ABL-webapp are the where you apply web server-level security
- URLs should always contain instance **and** service **and** transport
- Resources are optional, depending on transport

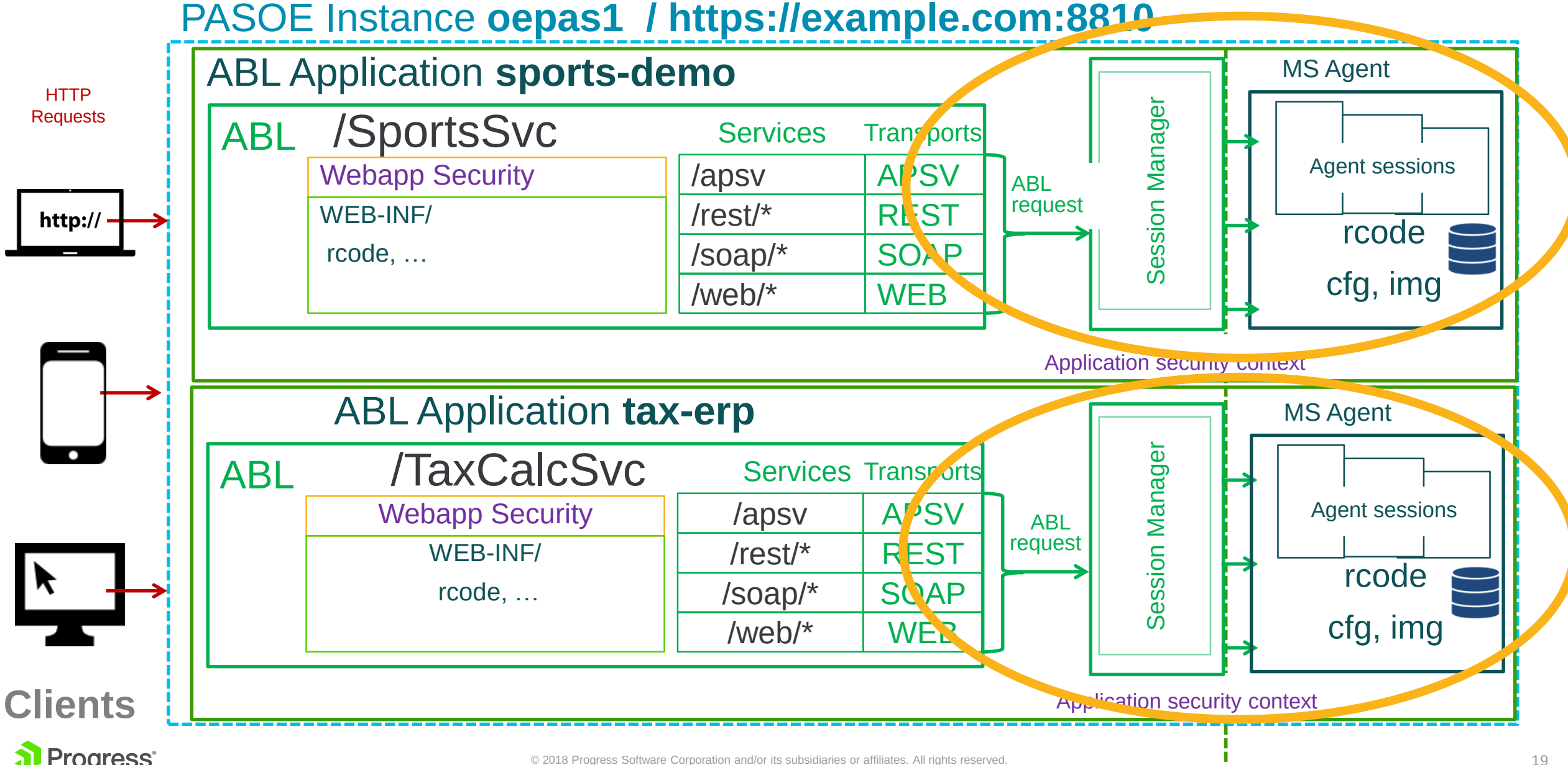
Why multiple ABL webapps?

- For ease of security configuration
  - 1 URL, 1 realm easy to grok
- To allow modularisation of application
  - Service interfaces
  - Add-on (paid) modules
  - Webapp traffic monitoring, billing, etc
- Acts as deployment package
  - Rcode (and source, optionally)
  - Configuration information
  - Database stuff (data, schema)
  - Patches/updates
- All of above could be done in a single ABL webapp, but way more complex

Instances, web-apps and ABL applications



Instances, web-apps and ABL applications



Why multiple instances & ABL apps?

- 1 PASOE instance ←→ 1 ABL application
  - Max scalability, flexibility
- 1 PASOE instance ←→ *n* ABL applications
  - A family of cooperating ABL applications from single vendor in same instance
  - Developer environment
  - Small deployments (limited resources: physical & people)
- Created using standard tcman tool

```
tcman.bat deploy $DLC/servers/PASOE/extras/oeabl.war \
      TaxCalcSvc    \      # web-app / URI
      tax-erp              # ABL app name
```

- Base ABL application location

```
$CATALINA_BASE/ablapps/<abl-app-name>
```

- Configuration in standard file/location

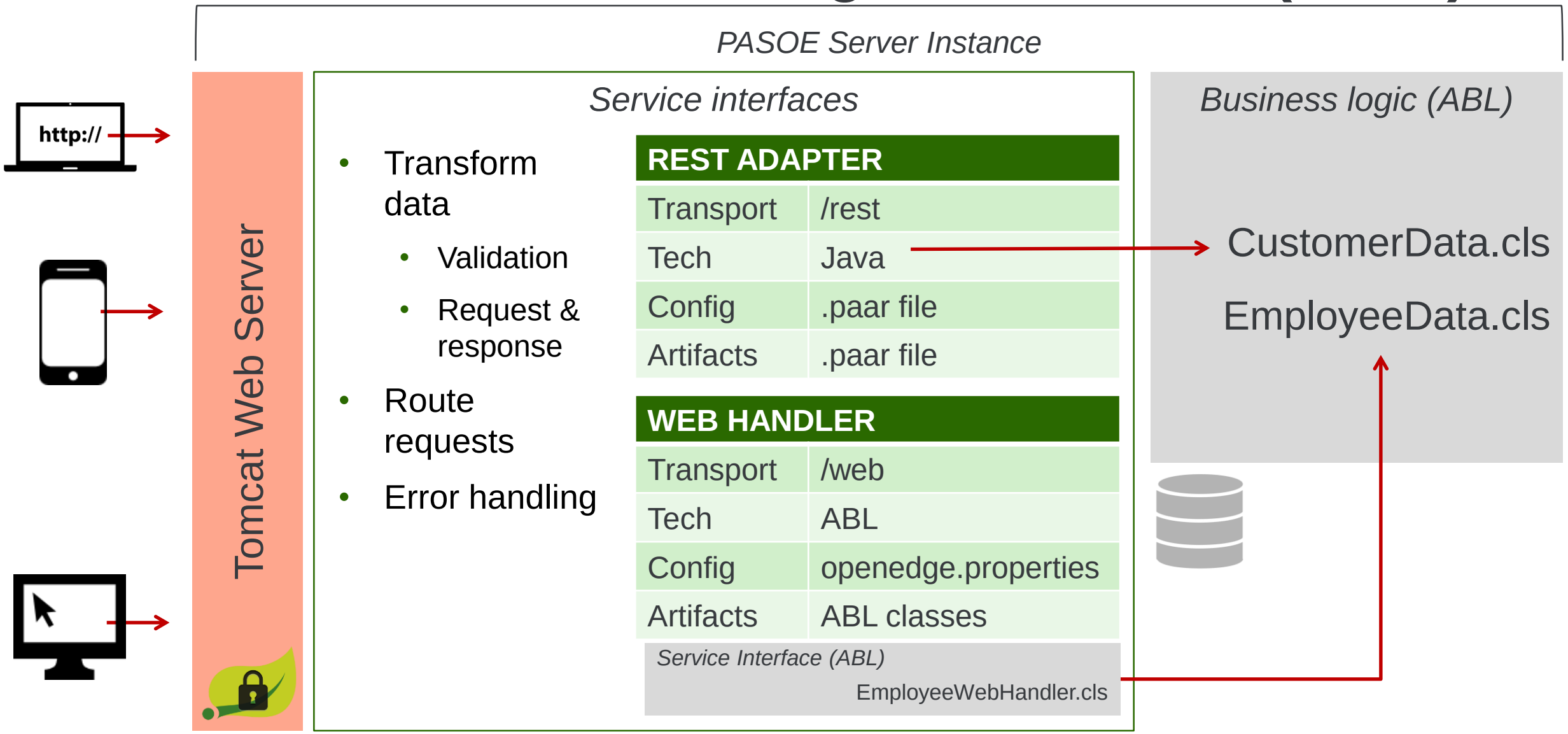
```
$CATALINA_BASE/conf/openedge.properties
```

Folder structure

- $CATALINA_BASE/conf contains almost all of the instance's configuration
  - $(oepas-app}/conf/oeablSecurity.* for security configuration
  - ${oepas-webapp}/WEB-INF/ /oeablSecurity.* for webapp-specific security configuration
- /openedge folders added to PROPATH
  - Modify it as needed

```
oepas1                                                  //${CATALINA_BASE}
+---ablapps
|   +---sports-demo                                     //${oepas-app}
|    |   +---conf
|    |   +---openedge
|    |    |   \---sports
|    |    |       EmployeeBE.cls
|    |   +---tlr
|   +---tax-erp                                         //${oepas-app}
+---bin
+---conf
        openedge.properties
+---logs
+---openedge
|   \---base
|           XmlUtil.cls
+---temp
+---webapps
|   +---TaxCalcSvc                                      //${oepas-webapp}
|   +---SportsSvc                                       //${oepas-webapp}
|   |   +---META-INF
|   |   +---static                                      //${oepas-ablsvc}
|   |   \---WEB-INF
|   |       +---adapters
|   |       |   +---rest                                //${oepas-ablsvc}
|   |       |   +---soap
|   |       +---openedge
|   |       |   \---sports
|   |       |           ImageWebHandler.cls             //${oepas-ablsvc}
|   |       \---tlr
\---work
```

Service Interface Technologies (11.6+)



Routing requests

HTML

```
GET /web/schedule HTTP/1.1
Host: pugchallenge.org
Accept: application/json


DELETE /web/schedule HTTP/1.1
Host: pugchallenge.org
Accept: application/json


POST /web/register HTTP/1.1
Host: pugchallenge.org
Content-Type: application/json


POST /web/feedback/session HTTP/1.1
Host: pugchallenge.org
Content-Type: application/json
```

ABL

Web.ScheduleHandler

provide_feedback.p

manage_sessions.p

register_attendee.p

Web.FeedbackHandler

Routing requests

HTML

```
GET /web/schedule HTTP/1.1
Host: pugchallenge.org
Accept: application/json


DELETE /web/
Host: pugcha
Accept: appl


POST /web/re
Host: pugcha
Content-Type


POST /web/feedback/session HTTP/1.1
Host: pugchallenge.org
Content-Type: application/json
```

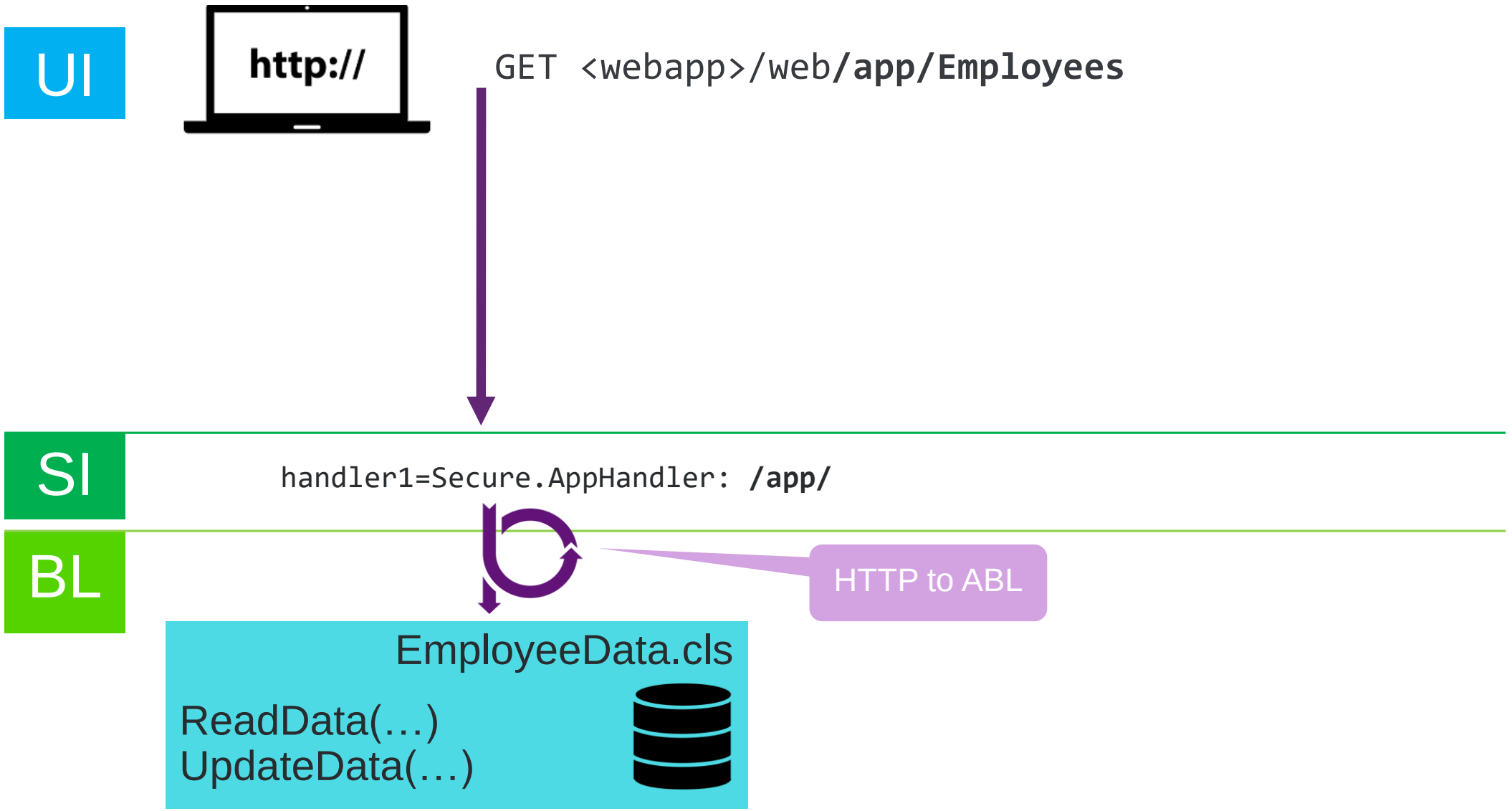
ABL

Web.ScheduleHandler

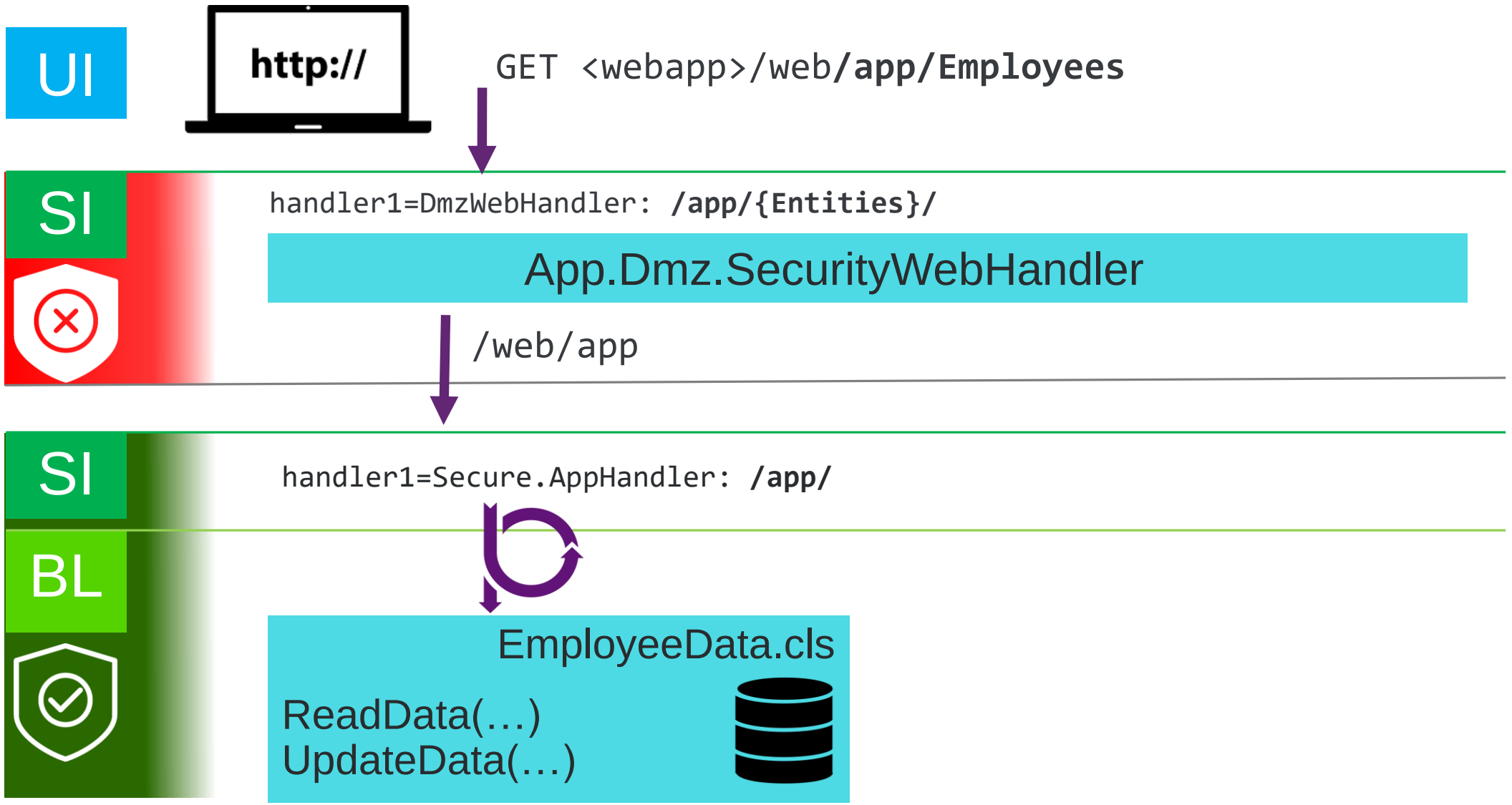
```
[pugpas.ConfApp.WEB]
    defaultHandler = OpenEdge.Web.CompatibilityHandler     // Classic WebSpeed
    handler1       = Web.ScheduleHandler          : /schedule/
    handler2       = Web.FeedbackHandler          : /feedback/session/{id}
    handler3       = Web.FeedbackHandler          : /feedback/session/
```

Web.FeedbackHandler

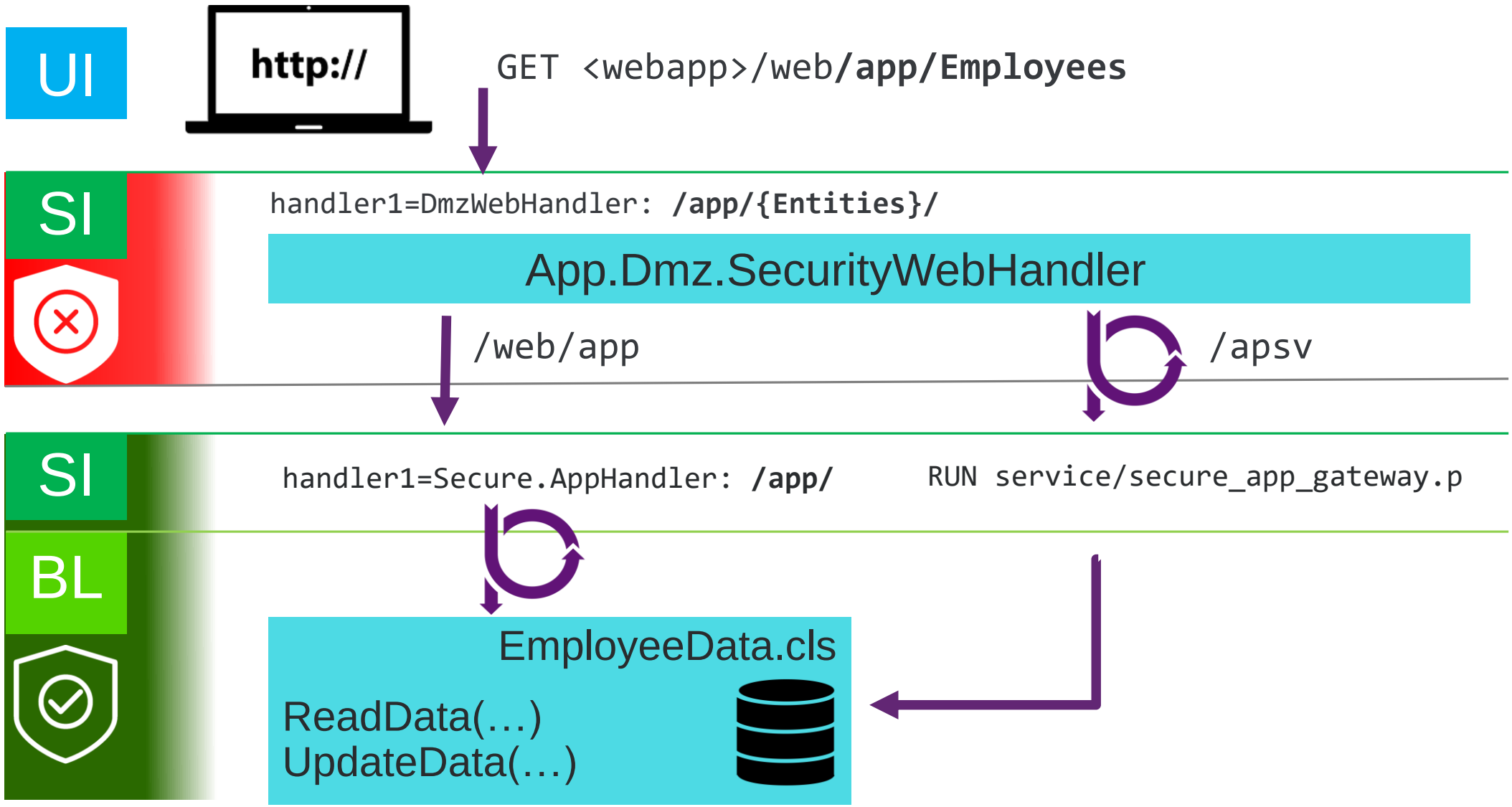
Single-tier architectures



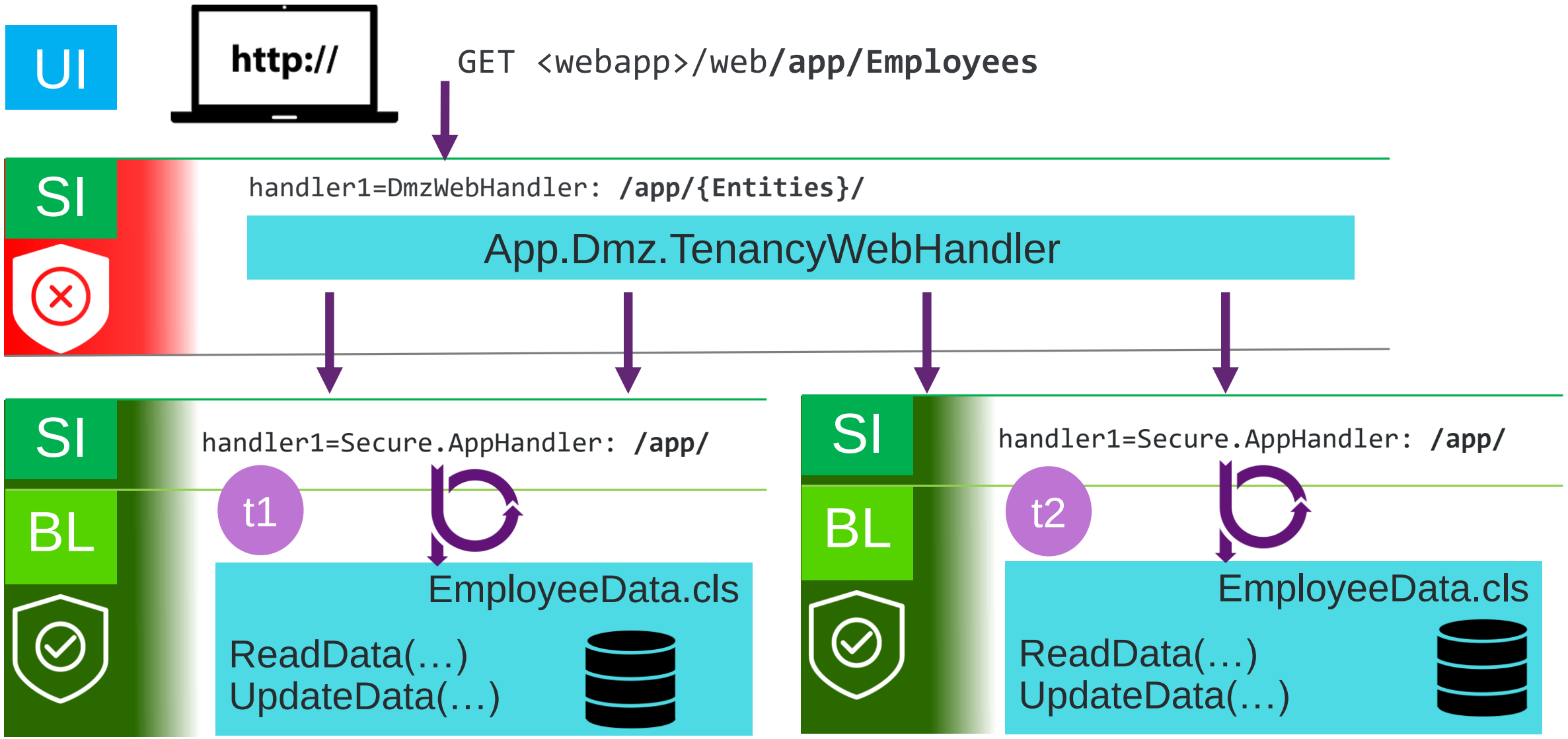
Multi-tier architectures



Multi-tier architectures



Multi-tier, multi-tenant architectures



Progress®

Monitoring

Resource allocation in PASOE

- We optimise for # agents, # sessions
- Process (agent) start is expensive
- Session start is expensive
  - Maximise reuse of sessions
  - OS-level monitoring of the agent process may not provide (much) insight
- At the OS level, thread-context switching is faster than process context switching
- We don't spread the load just for fun: if 1 session can deal with an entire application's load, it will
- Tomcat is master for concurrent request control

Monitoring PASOE instances

- PASOE was developed with industry standard monitoring in mind
- Built into a Tomcat container it allows for JMX monitoring in the JVM
  - Several available Tomcat monitoring tools available
  - Connects via port or PID
- REST APIs added for easy access of PASOE
  - Documented for easy use
  - Easily accessible by many tools (we include one)
  - Accessed via the same HTTP/S port as the application
  - But since it is a separate web application we can protect it

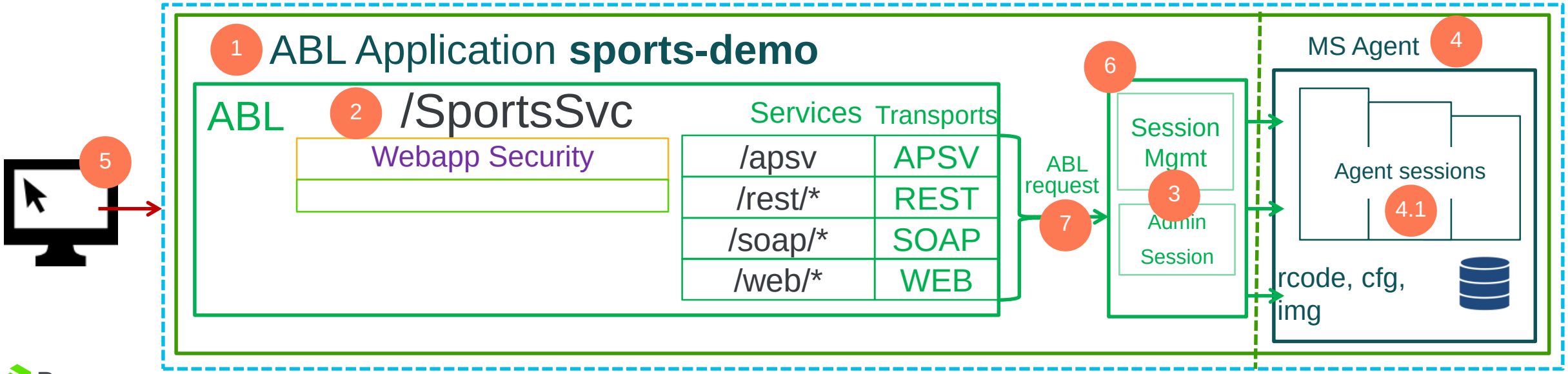
- OpenEdge Management
  - Has its own REST APIs for extensibility
  - Uses PASOE REST APIs

Rule #1: Always monitor

1. REST APIs for PASOE
   - Accessed via the same HTTP/S port as the application
   - But since it is a separate web application we can protect it
2. JMX calls to the Tomcat JVM
   - There are 2 ways to access the Tomcat JVM
     - Via the PID of the java process
     - Via network ports
   - There are 3 ways to gather information
     - Jconsole – a GUI based tool
     - Java code to the JVM
     - Jmxquery – A batch script

oemanager REST API

1. /oemanager/applications/
2. /oemanager/applications/<abl-app>/webapps
3. /oemanager/applications/<abl-app>/sessions
4. /oemanager/applications/<abl-app>/agents
    1. /oemanager/applications/<abl-app>/agents/<agent-id>/sessions
5. /oemanager/applications/<abl-app>/clients
6. /oemanager/applications/<abl-app>/metrics
7. /oemanager/applications/<abl-app>/requests



6

REST API

GET DELETE

`/oemanager/applications/<abl-app>/metrics`

- Think of metrics as `asbman -query`
  - concurrentConnectedClients
  - maxQueueDepth
  - maxConcurrentClients
  - readErrors
  - reserveConnectionTimeouts
  - timesQueued
  - avgQueueDepth

3

REST API

GET DELETE

/oemanager/applications/<abl-app>/sessions

```
"elapsedTimeMs": 3343,
  "bound": false,
  "requestState": "RUNNING",
  "requestID": "",
  "agentConnInfo": {
    "state": "RESERVED",
    "localAddr": "/127.0.0.1:60464",
    "agentID": "AG-/5vs3eApTMiPV5M+W47T3w",
    "connID": "AC-nRJgbuKtSNWcy6YWQknTbg",
    "connPoolID": "CP-30BInu3FRwOGQB0u0LwJ/g",
    "agentAddr": "localhost/127.0.0.1:62004"
  },
  "ablSessionID": "5",
  "sessionState": "RESERVED", // IDLE,ACTIVE
  "agentID": "AG-/5vs3eApTMiPV5M+W47T3w",
  "clientConnInfo": {
    "elapsedTimeMs": 5214,
    "requestProcedure": "ngasPing.p",
     "requestUrl": "http://localhost:16680/apsv",
     "clientName": "172.21.75.148",
     "httpSessionId": "<session-id>.pas1",
     "executerThreadId": "catalina-exec-8",
     "reqStartTimeStr": "2014-09-17T08:41:48.138-0400",
      "requestID": "ngasPing.p",
      "adapterType": "APSV", // REST,SOAP,WEB
      "sessionID": "Iphz68lUQKexuHibyy6S+A"
  },
  "adapterType": "APSV",
  "sessionPoolID": "SP-RFplWI9/Rta9j1pp+WcRyQ",
  "sessionID": "Iphz68lUQKexuHibyy6S+A",
  "sessionType": "SESSION_MANAGED", // SESSION_FREE
  "lastAccessStr": "2014-09-17T08:41:50.006-0400"
```

4.1

REST API

GET DELETE

/oemanager/applications/<abl-app>/agents/<agent-id>/sessions/


```
 "AgentSession": {
   "SessionId": 1,
   "SessionState": "ACTIVE",
   "StartTime": "2014-09-22T11:07:49.744",
   "EndTime": null,
   "ThreadId": 2,
   "ConnectiondId": 1342,
   "SessionExternalState": 0,
   "SessionMemory": 978433
 }
```

7

REST API

GET DELETE

/oemanager/applications/<abl-app>/requests

```
{
  "requestElapsedTime": 1889,
  "requestStartTimeStamp": "2016-09-17T10:11:26.934-0400",
  "requestState": "RUNNING",
  "requestID": "xfJqPACGU4jqE4pd0Auzwg",
  "sessionID": "8DvBtOplRHijAj7NcSAQKw"
}
```

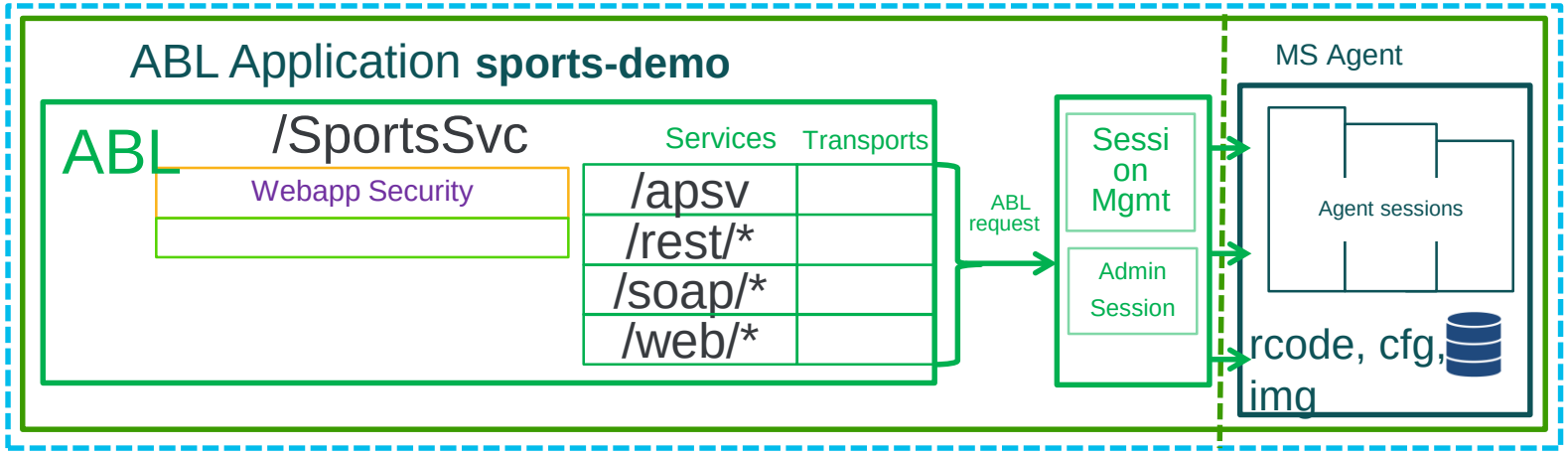
Progress®

Troubleshooting & logging

Troubleshooting

- Breakup the troubleshooting into
  - a) handling client requests (Tomcat instance)
  - b) executing ABL requests

- Where to look?
  - Tomcat
  - WebApp
  - Session Manager
  - Agent
  - ABL Session



The flow of a request

- Tomcat receives request
- Routes to webapp
- Routes to transport
- Routes to Session Manager
- Finds open connection (the Session Manager knows which agents have open connections)
  - Connection = available worker thread in an agent that contains an avm
  - Connection is the bridge between the webapp and the avm process
    (an open connection is a worker thread , NOT a session)
  - Session manager
    - Has list of agents (Ordered by time of instantiation)
      - Agent has list of connections (with status)
- Finds an available session
  - Worker thread looks in the agent's pool of sessions
- Executes ABL

Logging is your friend

- The service logging configuration found in
  `<web-app-name>/WEB-INF/logging.xml`
  `<project>/PASOEContent/WEB-INF/logging.xml`
- `DEBUG` and `TRACE` level produce lots of logging

monitor log file sizes

- LOG-MANAGER for ABL-related logging

```
<!-- OEABL Security -->
    <logger name="com.progress.appserv.services.security" level="INFO"/>

<!-- Spring Security -->
    <logger name="org.springframework" level="INFO"/>
    <logger name="org.springframework.security.web.access" level="INFO"/>
    <logger name="org.springframework.security.saml" level="INFO"/>

<!-- SAML -->
    <logger name="org.opensaml" level="INFO"/>
    <logger name=" PROTOCOL_MESSAGE " level="INFO"/>
```

Logging configuration

- Agent logfile rollovers

```
agentLogFile=

Location, name, and roll-over of the agent log file.
To have the agent log file rollover at midnight add {yyyy-
MM-dd} for the same format as other log files.
Example:  <path-to>/<instancename>.agent.{yyyy-MM-dd}.log
```

- Logfile archives

```
tcman clean -A
```

Progress®

Security

Securing the oemanager webapp

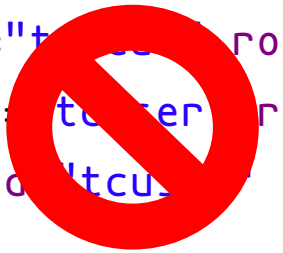
- Deploy the REST API application

```
<instance>/bin/tcman deploy $DLC/servers/pasoe/extras/oemanager.war
```

- Limit access to this application to only localhost requests
  - Add filter to **<instance>/webapps/oemanager/WEB-INF/web.xml**
- Access control
  - Change the default passwords for access

    <instance>/conf/tomcat-user.xml
  - Remove the default & sample users and password
  - Add new user(s) and password(s)

```
<user username="tomcat" password="t      roles="ROLE_PSCAdmin,ROLE_PSCOper,ROLE_PSCUser" />
<user username="tcuser1" password=      roles="ROLE_PSCUser" />
<user username="tcuser2"  password=      roles="ROLE_PSCNone" />
```



Securing the oemanager webapp

```
    <!-- END:container.security -->
    <filter>
        <filter-name>Remote Address Filter</filter-name>
        <filter-class>org.apache.catalina.filters.RemoteAddrFilter
        </filter-class>
        <init-param>
        <param-name>allow</param-name>
        <param-value>127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1</param-value>
        </init-param>
    </filter>

    <filter-mapping>
        <filter-name>Remote Address Filter</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>

</web-app>
```

Securing the ROOT application

The default ROOT webapp is the OpenEdge version (oeabl.war). If you are using multiple webapps and not ROOT …

- Rather use the Tomcat default version in $DLC/servers/pasoe/extras

```
<instance>/bin/tcman deploy $DLC/servers/pasoe/extras/ROOT.war
```

- Alternatively use a secured/hardened webapp as ROOT
  - There are plenty of guides on how to secure a Tomcat webapp

Progress®

Peter Judge pjudge@progress.com