PUG CHALLENGE EXCHANGE AMERICAS

APACHECON North America

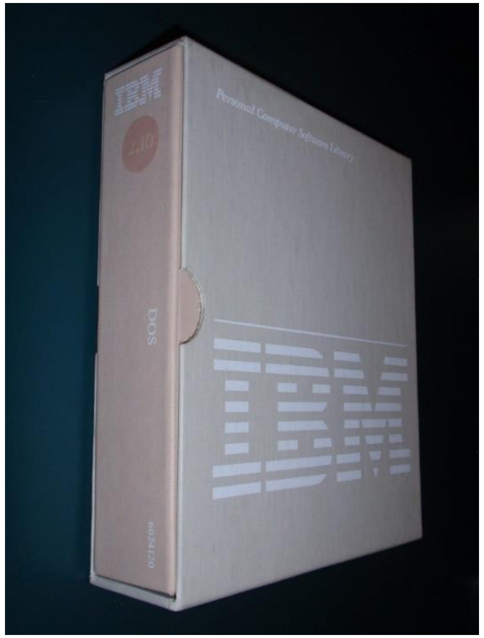
How I Stopped Worrying and Learned to Love Open Source

David Cleary

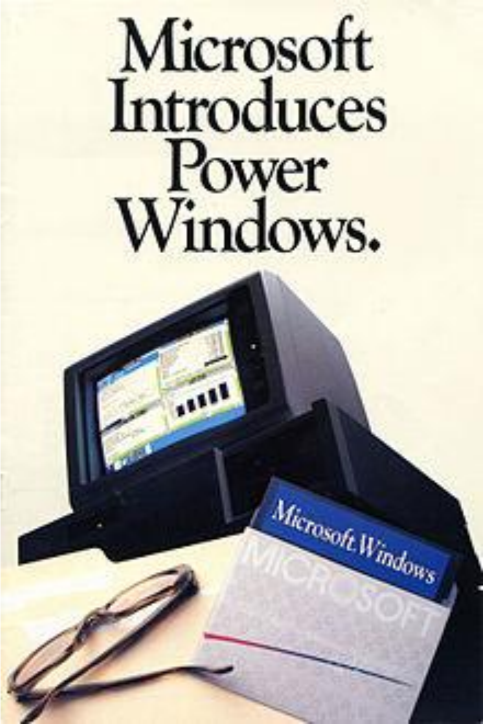
Progress

Progress Who?

Comdex 1983


IBM
DOS
IBM

IBM
128 KB_
IBM

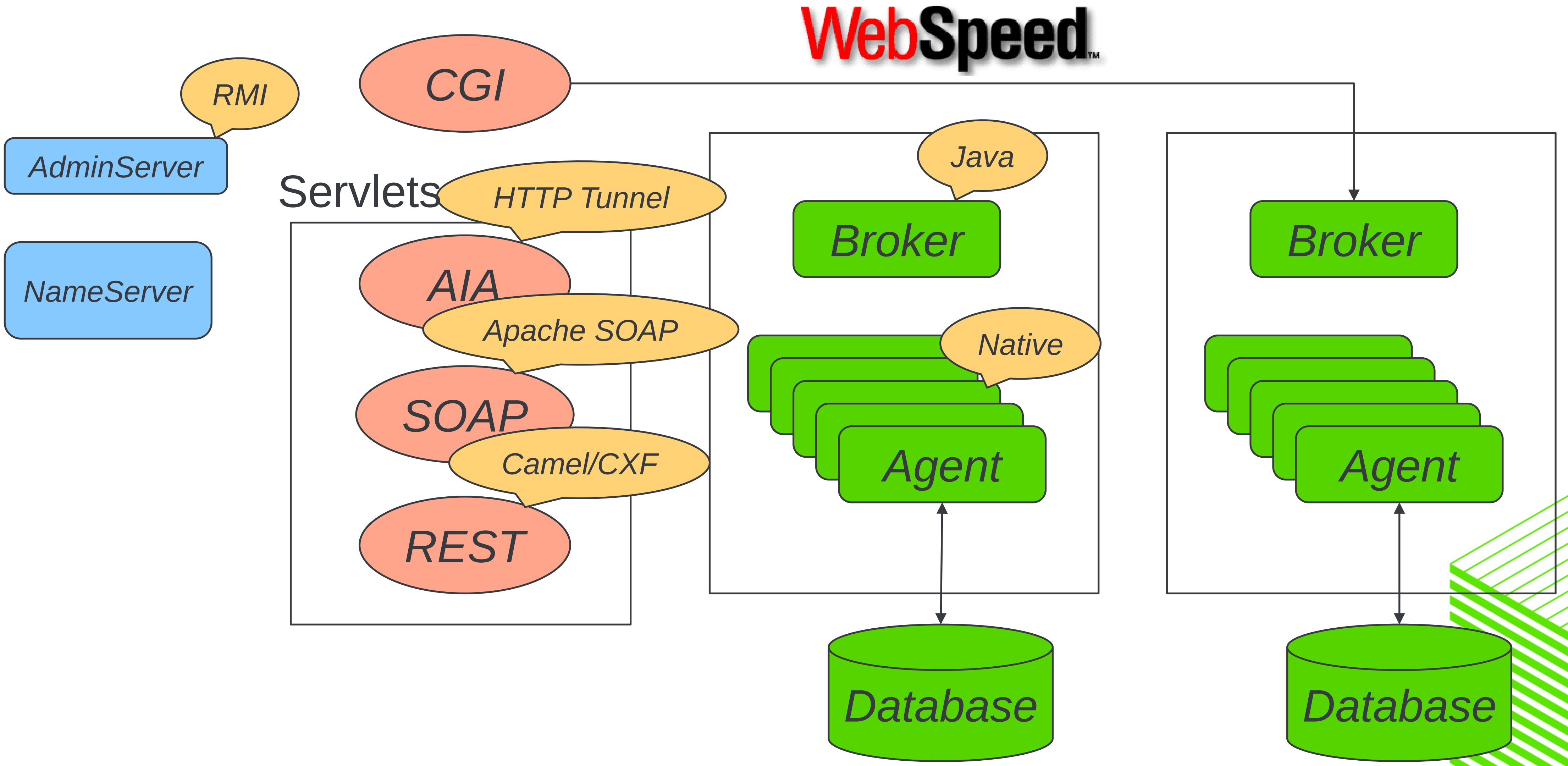

Microsoft Introduces Power Windows.
Microsoft Windows
MICROSOFT

GIANT
HALT AND CATCH FIRE

August 1984 – First Shipment of Progress 2.2

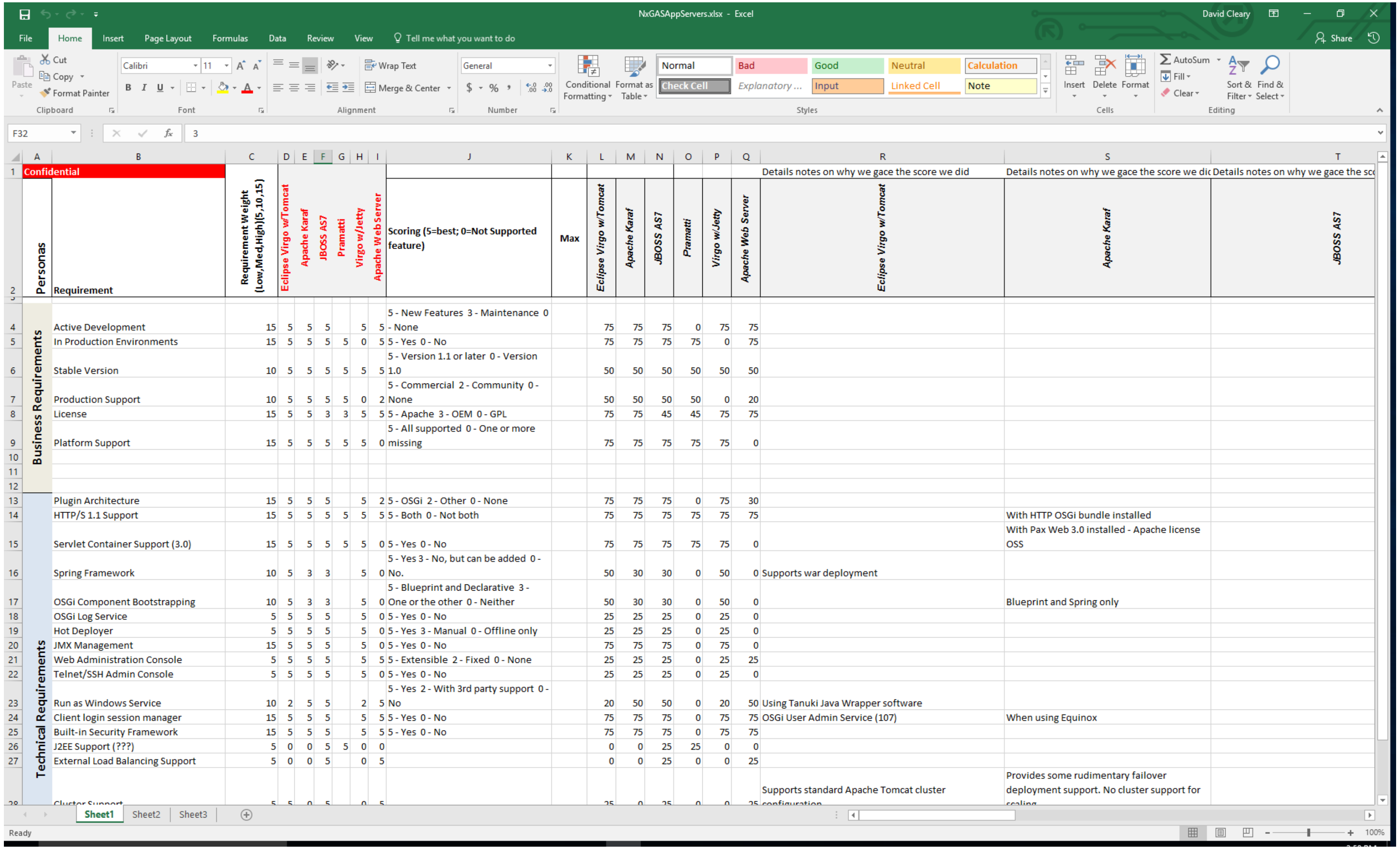
"Data Language Corp. has released Progress, a high-performance application development system. In use now on AT&T, Fortune Systems, and Convergent Technologies machines, Progress will soon be available for the IBM PC AT under MS-DOS and Xenix. Progress combines a powerful data base management system, application language, and an advanced user interface. Automatic screen and report generation, error recovery and an on-line tutorial are featured. Prices start at $ 1 ,450 for single users and $ 1 ,950 for multi-user systems. Query/run-time and plain run-time systems are available for sale with applications. A Progress Introductory System is available for $295, including on-line tutorial, full documentation, and all Progress facilities for building a working application limited only by data base size."

Progress Classic AppServer Architecture



Deciding on a Platform

Application Server Scorecard



F32 | 3

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	Confidential																	Details notes on why we gace the score we did	Details notes on why we gace the score we di	Details notes on why we gace the sc
2	Personas	Requirement	Requirement Weight (Low,Med,High)(5,10,15)	Eclipse Virgo w/Tomcat	Apache Karaf	JBOSS AS7	Pramatti	Virgo w/Jetty	Apache Web Server	Scoring (5=best; 0=Not Supported feature)	Max	Eclipse Virgo w/Tomcat	Apache Karaf	JBOSS AS7	Pramatti	Virgo w/Jetty	Apache Web Server	Eclipse Virgo w/Tomcat	Apache Karaf	JBOSS AS7
4	Business Requirements	Active Development	15	5	5	5		5	5	5 - New Features 3 - Maintenance 0 - None		75	75	75	0	75	75			
5		In Production Environments	15	5	5	5	5	0	5	5 - Yes 0 - No		75	75	75	75	0	75			
6		Stable Version	10	5	5	5	5	5	5	5 - Version 1.1 or later 0 - Version 1.0		50	50	50	50	50	50			
7		Production Support	10	5	5	5	5	0	2	5 - Commercial 2 - Community 0 - None		50	50	50	50	0	20			
8		License	15	5	5	3	3	5	5	5 - Apache 3 - OEM 0 - GPL		75	75	45	45	75	75			
9		Platform Support	15	5	5	5	5	5	0	5 - All supported 0 - One or more missing		75	75	75	75	75	0			
10																				
11																				
12																				
13	Technical Requirements	Plugin Architecture	15	5	5	5		5	2	5 - OSGi 2 - Other 0 - None		75	75	75	0	75	30			
14		HTTP/S 1.1 Support	15	5	5	5	5	5	5	5 - Both 0 - Not both		75	75	75	75	75	75		With HTTP OSGi bundle installed	
15		Servlet Container Support (3.0)	15	5	5	5	5	5	0	5 - Yes 0 - No		75	75	75	75	75	0		With Pax Web 3.0 installed - Apache license OSS	
16		Spring Framework	10	5	3	3		5	0	5 - Yes 3 - No, but can be added 0 - No.		50	30	30	0	50	0	Supports war deployment		
17		OSGi Component Bootstrapping	10	5	3	3		5	0	5 - Blueprint and Declarative 3 - One or the other 0 - Neither		50	30	30	0	50	0		Blueprint and Spring only	
18		OSGi Log Service	5	5	5	5		5	0	5 - Yes 0 - No		25	25	25	0	25	0			
19		Hot Deployer	5	5	5	5		5	0	5 - Yes 3 - Manual 0 - Offline only		25	25	25	0	25	0			
20		JMX Management	15	5	5	5		5	0	5 - Yes 0 - No		75	75	75	0	75	0			
21		Web Administration Console	5	5	5	5		5	5	5 - Extensible 2 - Fixed 0 - None		25	25	25	0	25	25			
22		Telnet/SSH Admin Console	5	5	5	5		5	0	5 - Yes 0 - No		25	25	25	0	25	0			
23		Run as Windows Service	10	2	5	5		2	5	5 - Yes 2 - With 3rd party support 0 - No		20	50	50	0	20	50	Using Tanuki Java Wrapper software		
24		Client login session manager	15	5	5	5		5	5	5 - Yes 0 - No		75	75	75	0	75	75	OSGi User Admin Service (107)	When using Equinox	
25		Built-in Security Framework	15	5	5	5		5	5	5 - Yes 0 - No		75	75	75	0	75	75			
26		J2EE Support (???)	5	0	0	5	5	0	0			0	0	25	25	0	0			
27		External Load Balancing Support	5	0	0	5		0	5			0	0	25	0	0	25			
28		Cluster Support	5	5	0	5		0	5			25	0	25	0	0	25	Supports standard Apache Tomcat cluster configuration	Provides some rudimentary failover deployment support. No cluster support for scaling	

Eclipse Virgo with Tomcat

- Reasons we chose Eclipse Virgo
  - Performance
  - OSGI architecture
  - Administration console
  - Spring integration
  - Built-in diagnostics
- Reasons we abandoned Eclipse Virgo
  - Difficulties getting legacy code to run
  - Pushback from other groups
  - Could no longer fight the server and meet deadline

PAS Architecture

- First and foremost : **IT IS Apache Tomcat** (initial 7.0.42 – current 8.5.11)
  - PSC may extend – but will **not** customize – the core Apache Tomcat server
  - Supports deployment of any Java / Tomcat compliant web application
  - PSC products may not create a dependency to use PAS
- PSC adds value to standard Tomcat
  - Simplified management [from automation scripts] of server.xml
  - Administrator friendly command line utility for common server tasks
  - Full support for Tomcat *instances*, including UNIX daemons and Windows services
  - Common location for shared 3rd party/PSC/ISV products across web applications
  - Drop in extensions to customize Tomcat's run-time environment (via setenv) for web apps
  - Drop in extensions to customize creation of Tomcat instances
  - Removes unsecure remote management and ROOT web application & distributes as *extras*
  - Predefined configuration of security and production grade Tomcat features

Preconfigured Apache Tomcat Features

- Authentication Realm plug-ins (local file, LDAP, JAAS, …)
- HTTP session management [with cluster support]
- Java security manager integration
- Multiple server *instance* support
- Filters for white/black list checking
- Logging
- Optional JMX console administration
- HTTPS, HTTP, and AJP13 (worker) connectors
- Tomcat SSO
- Session ID size (22)
- SSL Java keystore and test server certificate (self-signed)
- Web crawler session protection
- Memory leak monitoring

PSC Supplied 3rd Party Extensions

- Single, scriptable, command line tool (tcman) for most common server administration
- Spring Security and Spring MVC support
- Apache commons http client
- Spring Security authn: digest, file, LDAP, AD, OpenID, CAS, SAML2 (more to come)
- Externalized server.xml values to easy to maintain property files
- Externalized enable/disable of individual server.xml features
- Secure *ROOT* web application (blank web application)
- *Extras* directory for optional and standard tomcat artifact distribution
- Windows service

Managing PAS and PAS Instances

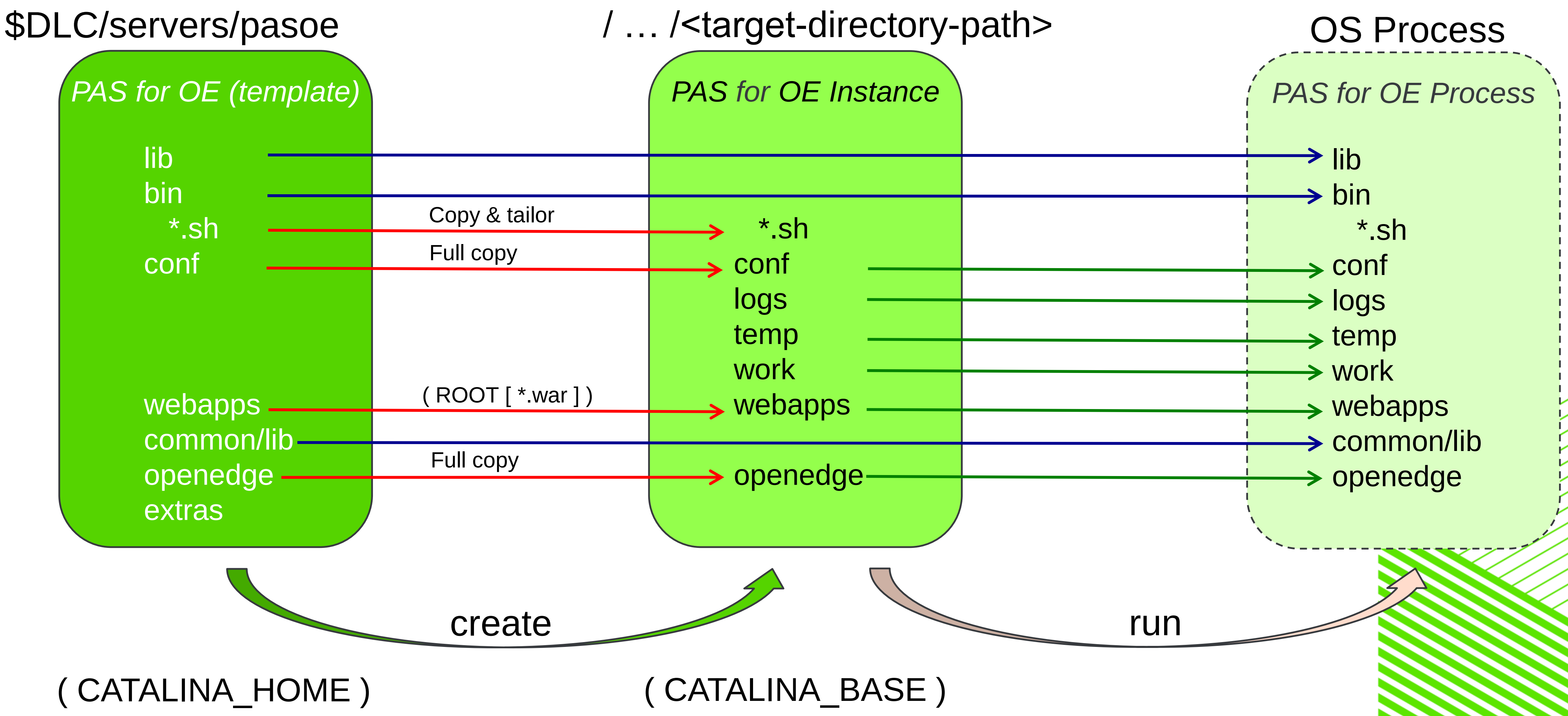
- PAS command line tool *tcman* (UNIX shell script & Windows Powershell)
- Manage each instance independently – Manage all instances from HOME PAS
- Records instances in HOME conf directory
- Each instance is assigned an *alias* name – doubles as JVM route for clusters
- Actions
    - List, Create, Delete
    - Register, Unregister
    - Workers.properties
    - Start, Stop, Test, Version
    - Config[uration]
    - Enable/Disable Tomcat features
    - Integration with Tomcat manger if installed

The Tomcat Instance Architecture

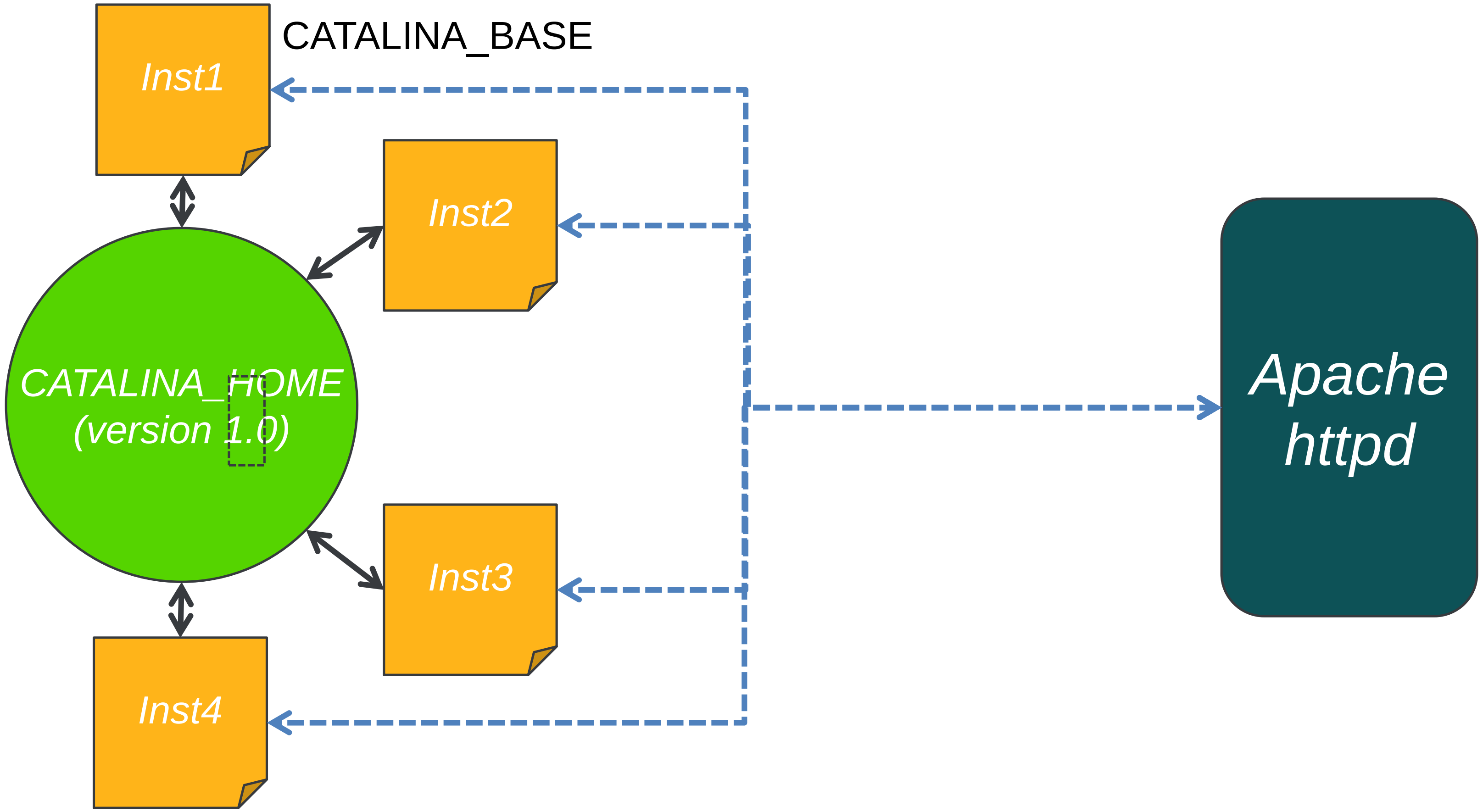
Tomcat *Instances* Offer More Architectural Options

- A run-time server configuration that shares common binaries, libraries, and scripts with the home server installation
- Each instance is a full Tomcat server process (with unique network ports)
- Lightweight expansion of the # of servers for load balancing and scaling
- Can have its own configuration and optionally its own set of deployed web applications
- Can have its own shared web application libraries
- Can be preconfigured and packaged as a deployable unit in ISV on-premise installations
- Lifetime can span multiple home PAS uninstalls and installs
- Updating the home PAS updates all instances
- Web application shared libraries can be updated without affecting any other server
- Can easily share web applications with other instances

Understanding PAS for OpenEdge Instance Run-time

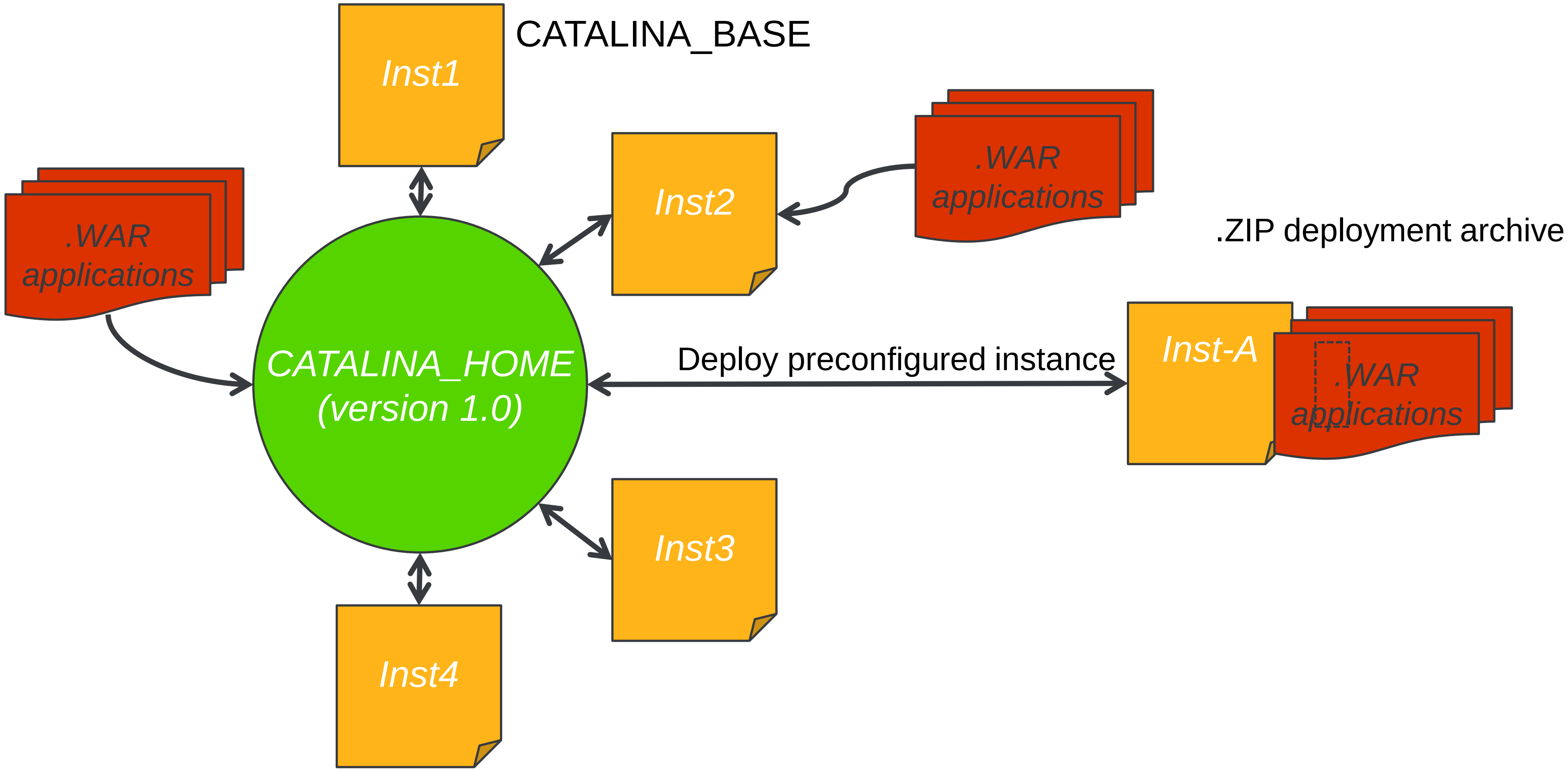


Instance Topology

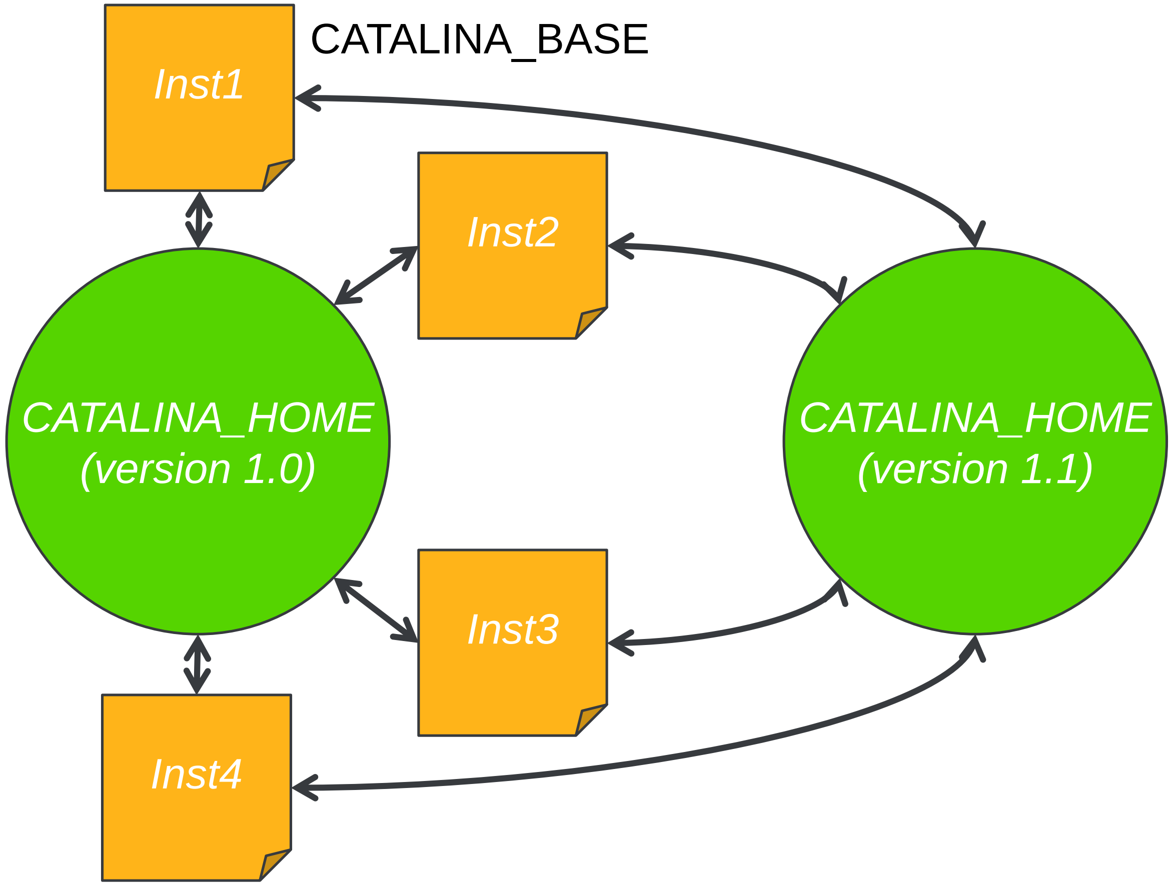


Tomcat Cluster

Instance Deployment



Upgrades Using Instances



Spring Security

Original Spring Security Configuration

- Required to manually edit XML files with hard-coded values
- Cannot be patched, updated, or hot-fixed
- 90% redundency between many files results in more testing, inconsistencies, & regressions
- No GUI tools to simplify local/remote administration
- The list of files is large, would only get larger
- High maintenance because common configuration properties not shared across web applications in the same ABL application
  (refer to the AppServer ubroker.properties layout)

Configuration Process Differences

11.6.x

- **Initial Development:**
  Edit web.xml – select one of *12 files*
  Edit *XML file* for each user account source
  Edit XML file for each URL access control
  (for REST & WEB transports)
- **Release testing:**
  Edit web.xml – for each: select file & test account logins to URLs and Methods
- **Upgrades, patches, … :**
  Edit-merge from OE distributed text document

11.7.x

- **Initial Development:**
  Edit *property file and select user account sources*
  *Edit **once** the csv file for URL access controls*
  (for ALL transports)
- **Release testing:**
  Edit *property file's user account source* & test account logins to URLs and Methods
- **Upgrades, patches, … :**
  Run OE upgrade/patch utility

Configuring Spring Security HTTP Request Filters & Login Account Sources

You Configure the Same Beans & Same Properties

11.6.x .XML file

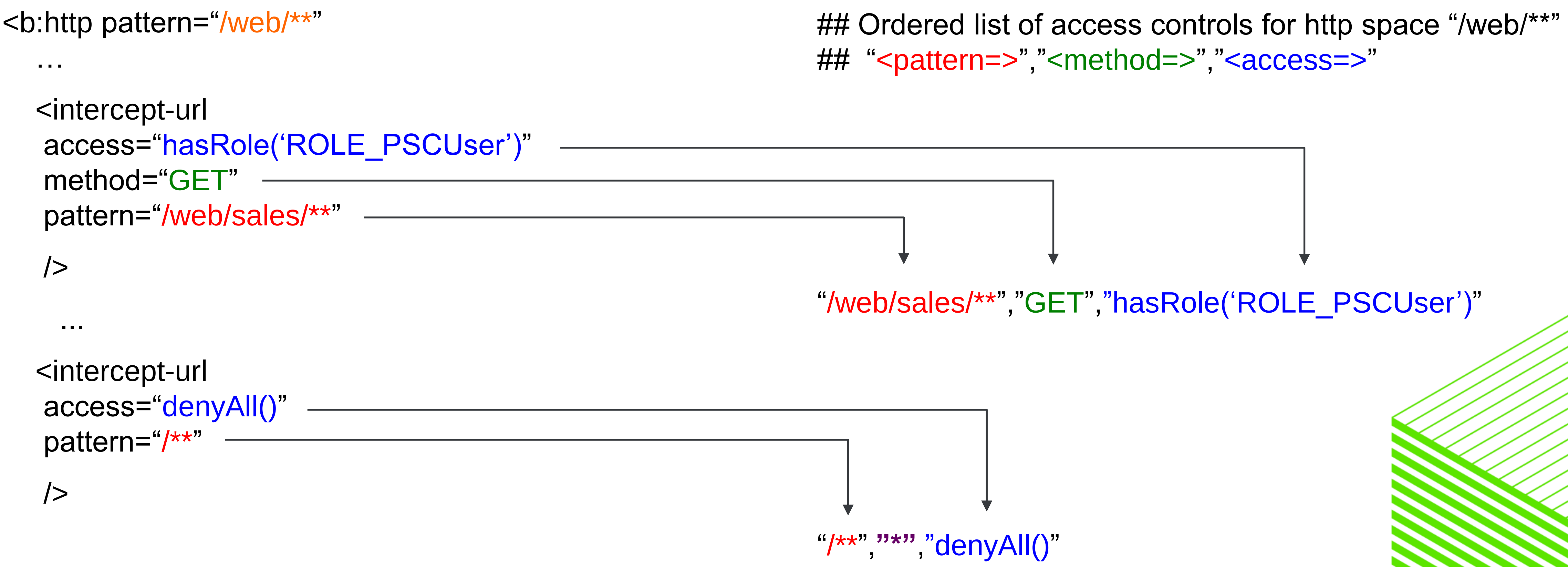
```
<b:bean id="OEClientPrincipalFilter"
    class="com.progress…OEClientPrincipalFitler">

   <b:property name="domain" value="<edited-value>" />

   <b:property name="key" value="<edited-value>" />

   <!– commented out properties
   b:property name="enablecp" value="<sample>" />
   b:property name="registryFile" value="<sample>" />
   b:property name="anonymous" value="<sample>" />
   b:property name="roleFilter" value="<sample>" />
             …
   -->

</b:bean>
```

11.7.x Property File

```
## <b:bean id="OEClientPrincipalFilter"

OEClientPrincpalFilter.domain=<edited-value>

OEClientPrincpalFilter.key=<edited-value>

## full list of properties & default values
OEClientPrincpalFilter.enablecp=true
OEClientPrincpalFilter.registryFile=
OEClientPrincpalFilter.anonymous=false
OEClientPrincpalFilter.roleFilter=""
             …
```

Configuring Spring Security URL Access controls (aka <intercept-url>)

You Configure the Same Intercept-url Access Controls



11.6.x .XML file

```
<b:http pattern=“/web/**”
    …
    <intercept-url
     access=“hasRole(‘ROLE_PSCUser’)”
     method=“GET”
     pattern=“/web/sales/**”
     />
      ...
    <intercept-url
     access=“denyAll()”
     pattern=“/**”
     />
```

11.7.x CSV File

```
## Ordered list of access controls for http space “/web/**”
##  “<pattern=>”,”<method=>”,”<access=>”

“/web/sales/**”,”GET”,”hasRole(‘ROLE_PSCUser’)”

“/**”,”*”,”denyAll()”
```

Use the Same Basic Guidelines for Web Application’s Access

- You Configure An Intercept-url control for
  - Each REST Service Interface or Business Entity (GET & POST methods only)
  - Each Web Web-Handler (only the methods supported by the ABL Web Handler class)
- Change the default to deny what is not explicitly granted
  from: “/web/**”,”*”,”hasRole(‘ROLE_PSCUser’)”
  to: “/web/**,”*”,”denyAll()”
- Order is IMPORTANT!!!
  - Fine grained URL patterns first, coarser grained URLs later
- The URL pattern matching is “ANT” – as in Apache ANT
  - A single wildcard (*) matches any filename/extension characters
  - A double wildcard (**) matches any set of directory & subdirectories
- Uses Spring Security’s Access Control Expressions
- A method may be a wildcard (*) for all methods, or a SINGLE method name

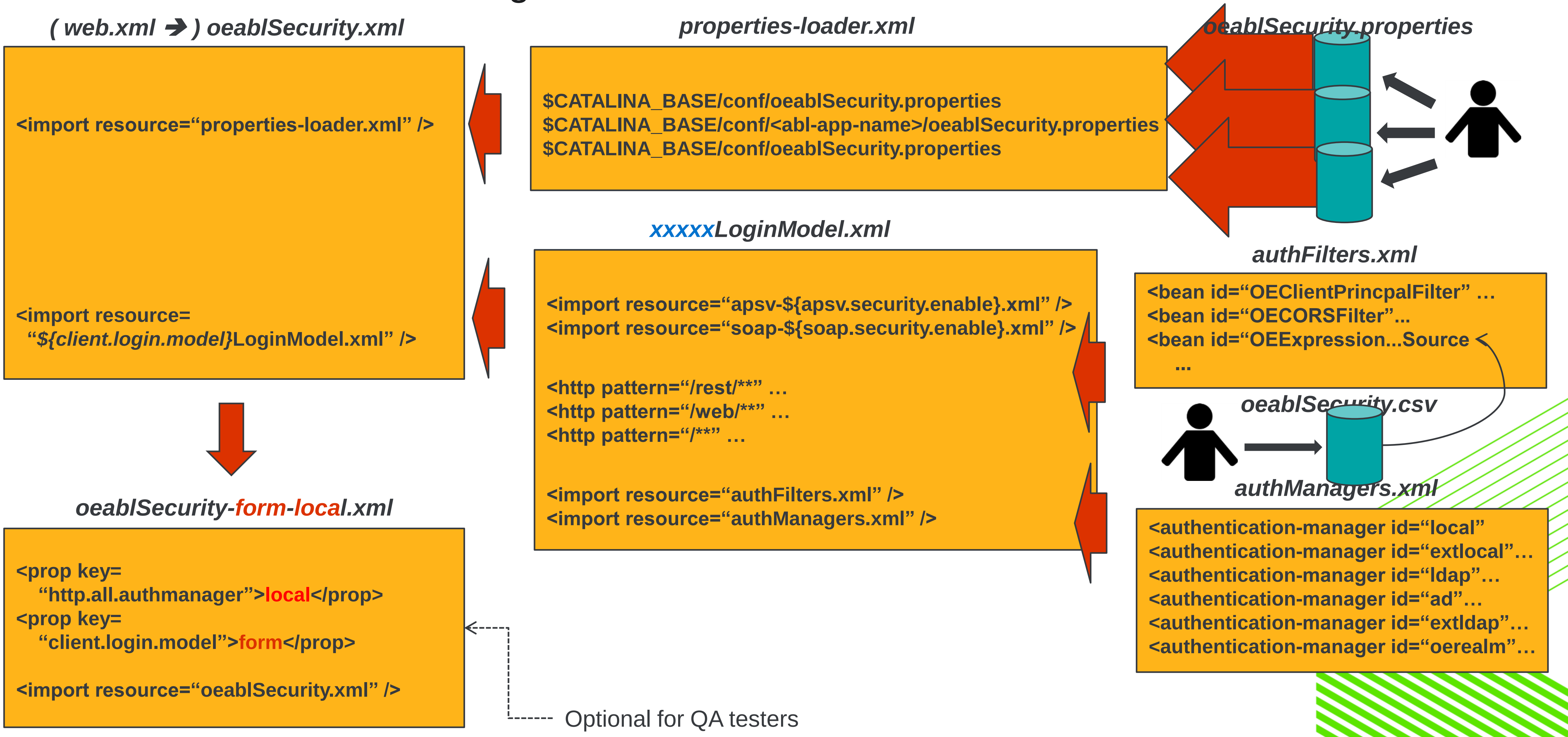
Layered Spring Security Configuration Property Files

1. webapps/<web-app-name>/WEB-INF/oeablSecurity.properties
   - Properties and values applied to the web application
   - <web-app-name> matches deployment configuration in conf/openedge.properties
   - Can contain all or subset of Spring Security properties
   - Supersedes property values defined in conf/

2. ablapps/<abl-app-name>/conf/oeablSecurity.properties
   - Defaults applied to all web applications within a single ABL business application
   - <abl-app-name> matches deployment configuration in conf/openedge.properties
   - Can contain all or subset of Spring Security properties
   - Supersedes property values defined in conf/oeablSecurity.properties

3. conf/oeablSecurity.properties
   - Superset of all Spring Security properties
   - Defaults applied to all web applications across all deployed ABL business applications

So How Does It All Fit Together At Run-time?



(web.xml ➔) oeablSecurity.xml

```
<import resource="properties-loader.xml" />

<import resource=
  "${client.login.model}LoginModel.xml" />
```

properties-loader.xml

```
$CATALINA_BASE/conf/oeablSecurity.properties
$CATALINA_BASE/conf/<abl-app-name>/oeablSecurity.properties
$CATALINA_BASE/conf/oeablSecurity.properties
```

oeablSecurity.properties

xxxxxLoginModel.xml

```
<import resource="apsv-${apsv.security.enable}.xml" />
<import resource="soap-${soap.security.enable}.xml" />

<http pattern="/rest/**" …
<http pattern="/web/**" …
<http pattern="/**" …

<import resource="authFilters.xml" />
<import resource="authManagers.xml" />
```

authFilters.xml

```
<bean id="OEClientPrincpalFilter" …
<bean id="OECORSFilter"…
<bean id="OEExpression…Source
    …
```

oeablSecurity.csv

authManagers.xml

```
<authentication-manager id="local"
<authentication-manager id="extlocal"…
<authentication-manager id="ldap"…
<authentication-manager id="ad"…
<authentication-manager id="extldap"…
<authentication-manager id="oerealm"…
```

oeablSecurity-form-local.xml

```
<prop key=
   "http.all.authmanager">local</prop>
<prop key=
   "client.login.model">form</prop>

<import resource="oeablSecurity.xml" />
```

Optional for QA testers

Selecting The Login Model & User Account Source in oeablSecurity.properties

- spring.login.model=

anonymous	# the default – no direct logins or SSO allowed
basic	# HTTP BASIC header direct logins & SSO headers
form	# HTTP (POST) form fields for direct login & SSO headers
container	# Tomcat realms integration & SSO headers
sso	# No direct login – only SSO headers

- http.all.authmanager=

local	# the application's users.properties (clear-text password)
extlocal	# the application's users.properties (encrypted passwords)
ldap	# simple LDAP (or Active Directory) server configuration
oerealm	# bridge to ABL application maintained user accounts
ad	# Simple (constrained) Active Directory configuration

Challenges

PSC Product Development … Challenges

- Same general challenges in sharing the same server with other web applications
- Logging – we have already seen where different web applications have issues
- JAR library hell –
  - Sharing libraries is good, but in Java it can be EVIL
  - Coordination of multiple PSC products using same library version
  - Using the Tomcat lib for general product libraries can cause server startup problems
  - Products are not required to use the PAS shared libraries or directory
- Multiple products installing their private version of the same file
- Product web applications that store temp/work data inside the web application

For more information

OpenEdge Developers Kit Classroom Edition

Progress® OpenEdge®

Includes fully functional PASOE Development Server

https://www.progress.com/openedge/classroom-edition

Progress® Corticon®

https://www.progress.com/corticon

Progress® Rollbase®

https://www.progress.com/rollbase

APACHECON
North America

PUGCHALLENGE
AMERICAS

EXCHANGE