IBM Washington Systems Center

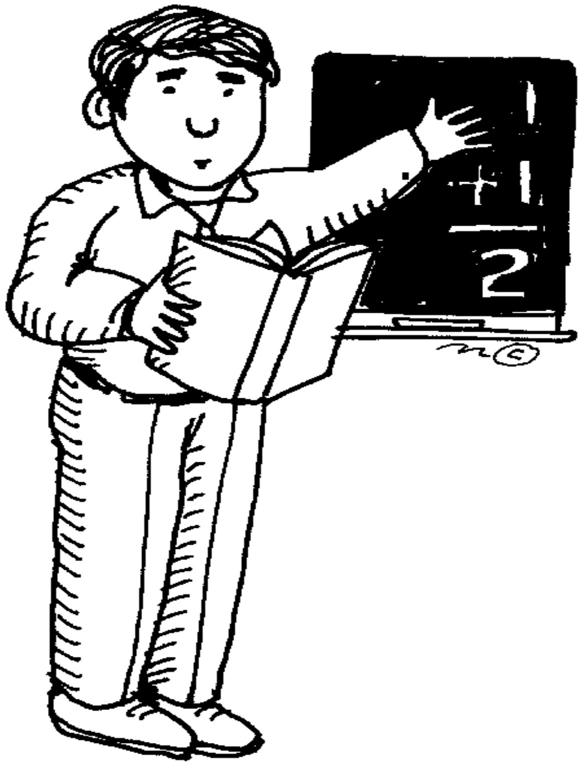
Application Management on WAS v6 for z/OS

Lee-Win Tai

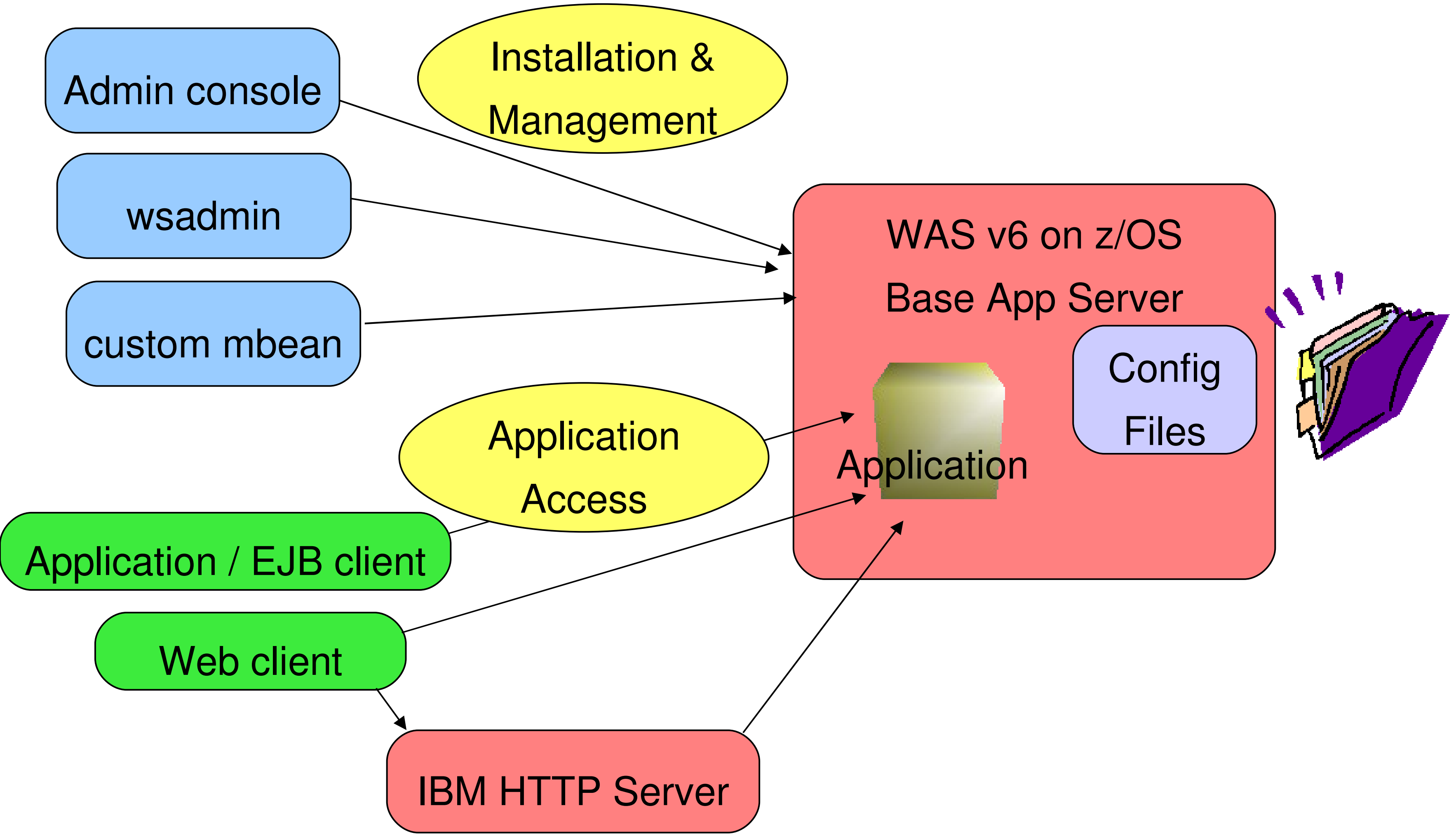
tai@us.ibm.com

Our Agenda

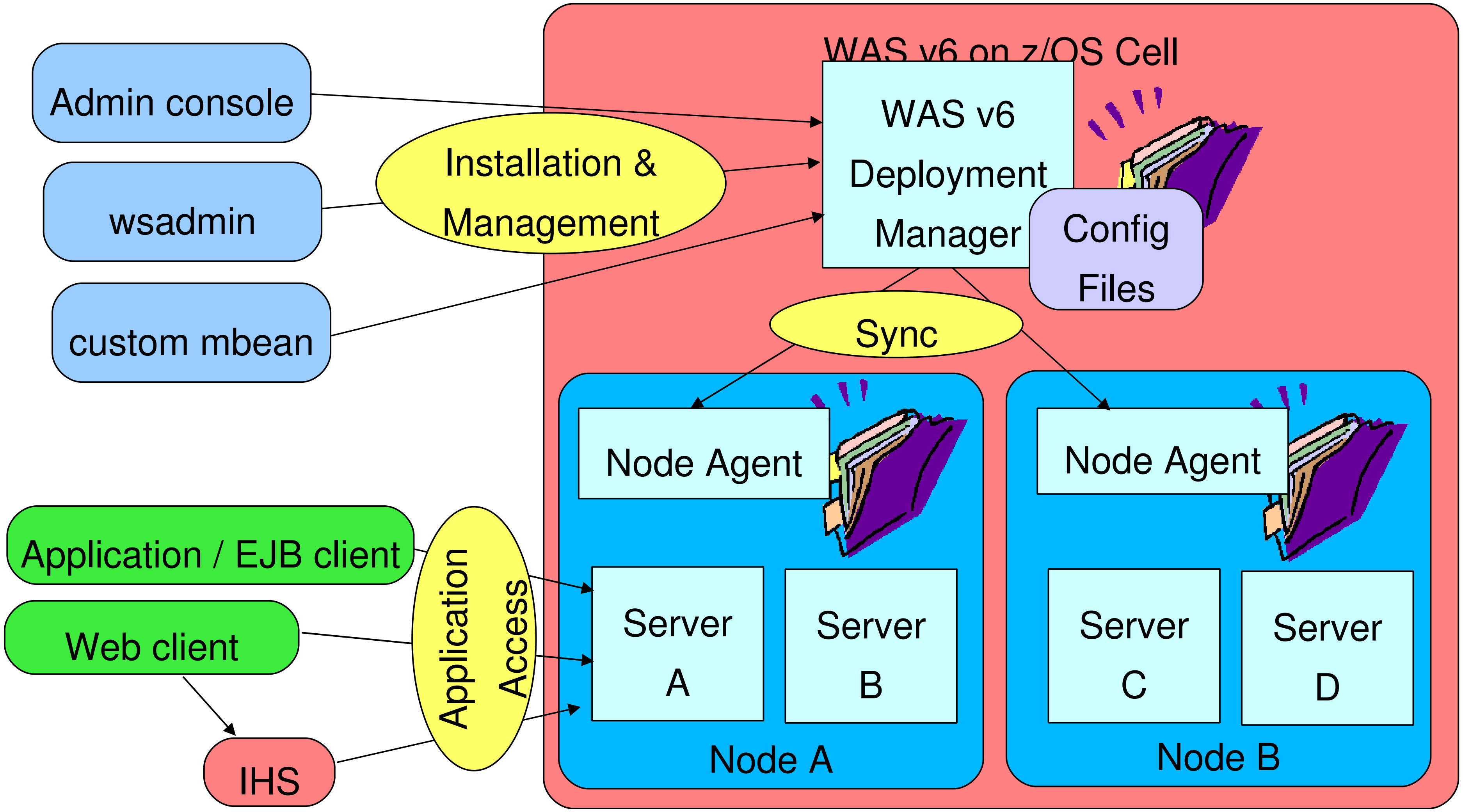
- **Administration Console**
- **Deploying Your Application**
- **Updating Your Application**
- **Application Scoped Resources**
- **Enhanced EAR**
- **WebSphere Rapid Deploymen**



Installing and Managing on a Base App Server



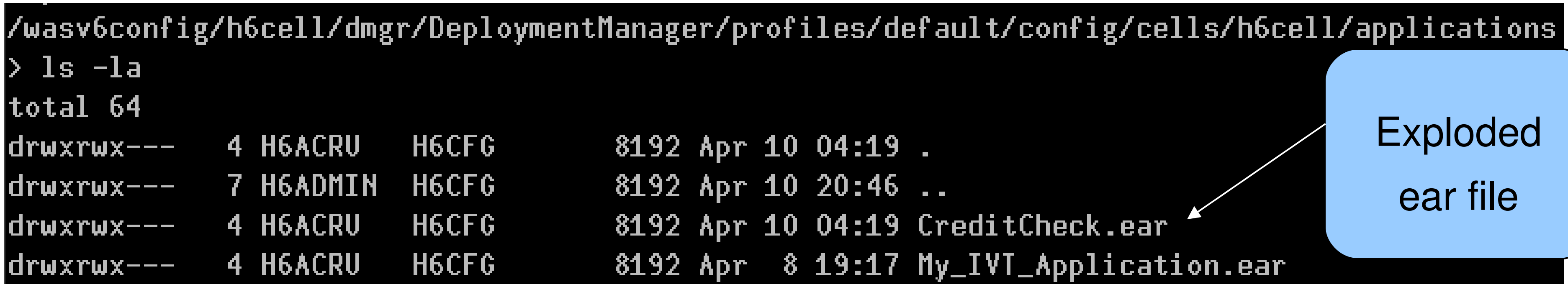
Installing and Managing in an ND Environment



Where Is My App?

Your application is installed at the cell level

Cell level directory structure

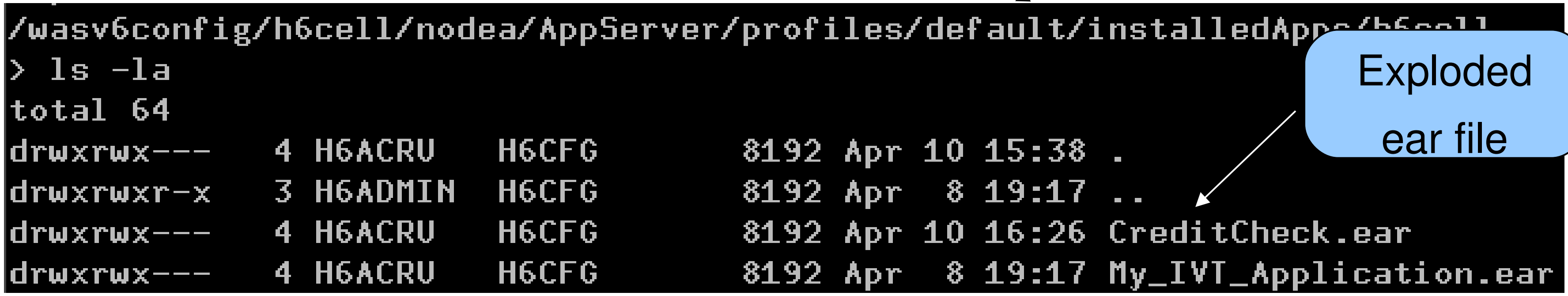


```
/wasv6config/h6cell/dmgr/DeploymentManager/profiles/default/config/cells/h6cell/applications
> ls -la
total 64
drwxrwx---    4 H6ACRU    H6CFG        8192 Apr 10 04:19 .
drwxrwx---    7 H6ADMIN   H6CFG        8192 Apr 10 20:46 ..
drwxrwx---    4 H6ACRU    H6CFG        8192 Apr 10 04:19 CreditCheck.ear
drwxrwx---    4 H6ACRU    H6CFG        8192 Apr  8 19:17 My_IVT_Application.ear
```

Exploded ear file

Then your application is mapped to the appropriate server

AppServer level directory structure



```
/wasv6config/h6cell/nodea/AppServer/profiles/default/installedApps/h6cell
> ls -la
total 64
drwxrwx---    4 H6ACRU    H6CFG        8192 Apr 10 15:38 .
drwxrwxr-x    3 H6ADMIN   H6CFG        8192 Apr  8 19:17 ..
drwxrwx---    4 H6ACRU    H6CFG        8192 Apr 10 16:26 CreditCheck.ear
drwxrwx---    4 H6ACRU    H6CFG        8192 Apr  8 19:17 My_IVT_Application.ear
```

Exploded ear file

New and Improved Admin Console

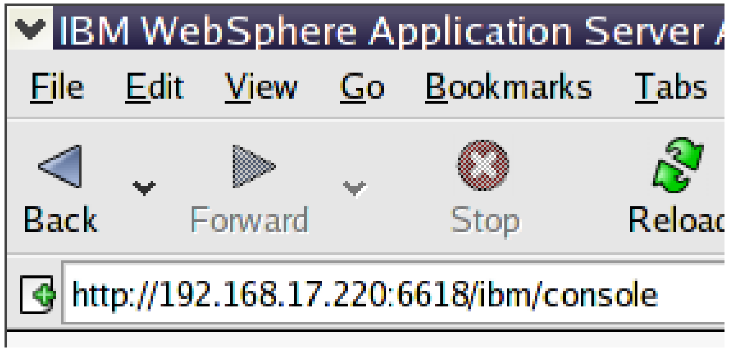
- New appearance
- Usability enhancements
- IBM HTTP server management
- Tivoli Performance Viewer integration

Less scrolling!

Discussed in next few slides

New Admin Console URL
http://server[:port]/ibm/console

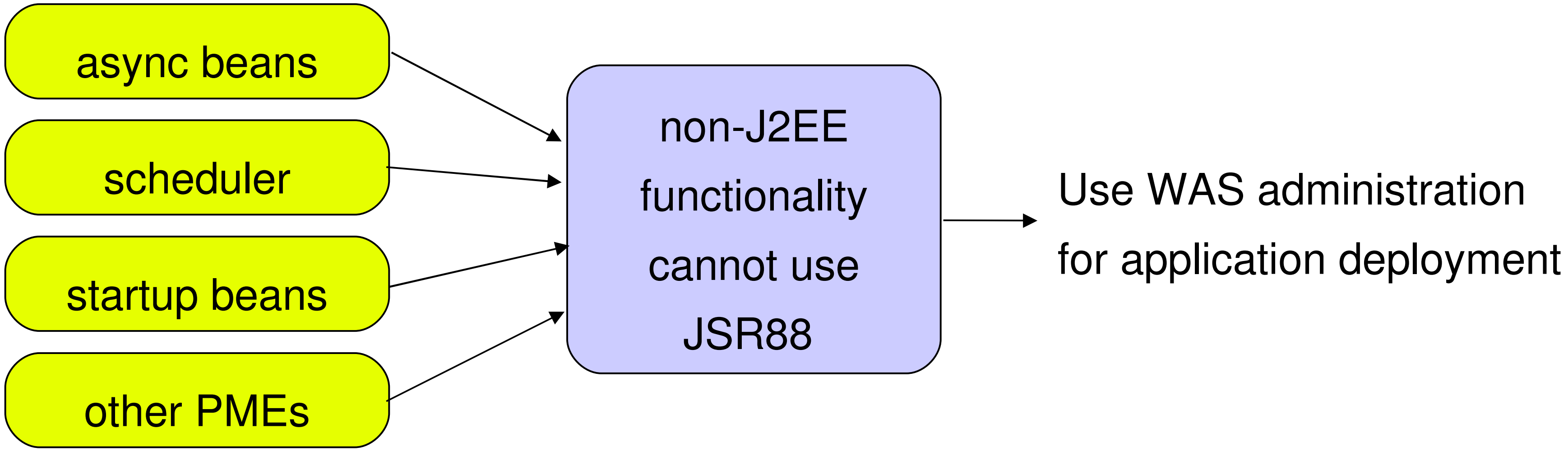
http://server[:port]/admin will redirect



http://192.168.17.220:6618/ibm/console

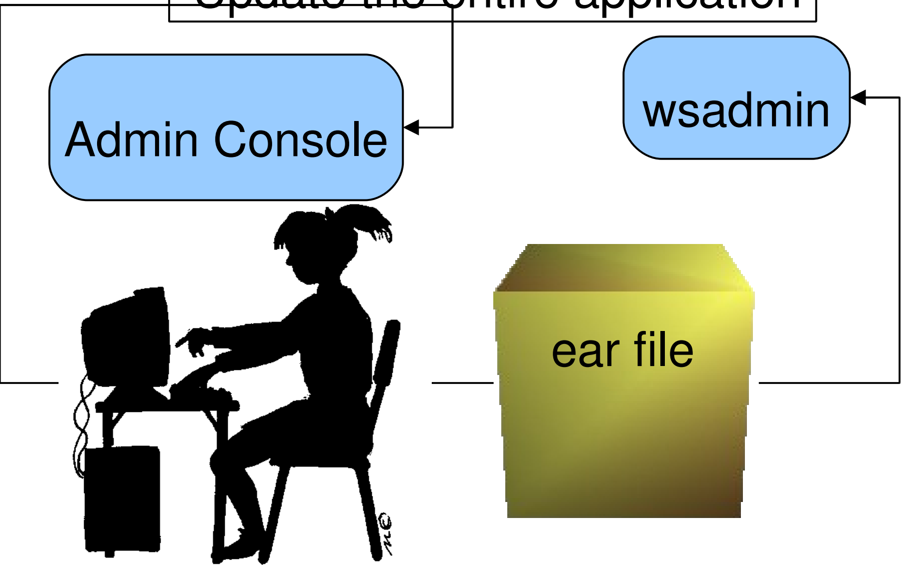
J2EE Support in WAS v6

- WAS v6 for z/OS supports J2EE 1.4, 1.3, and 1.2
- J2EE 1.4 compliant ear files can contain J2EE 1.2 and 1.3 modules
- Support for the Application Deployment API 1.1 – JSR 88
  - an alternative to the admin console, wsadmin, or custom mbean

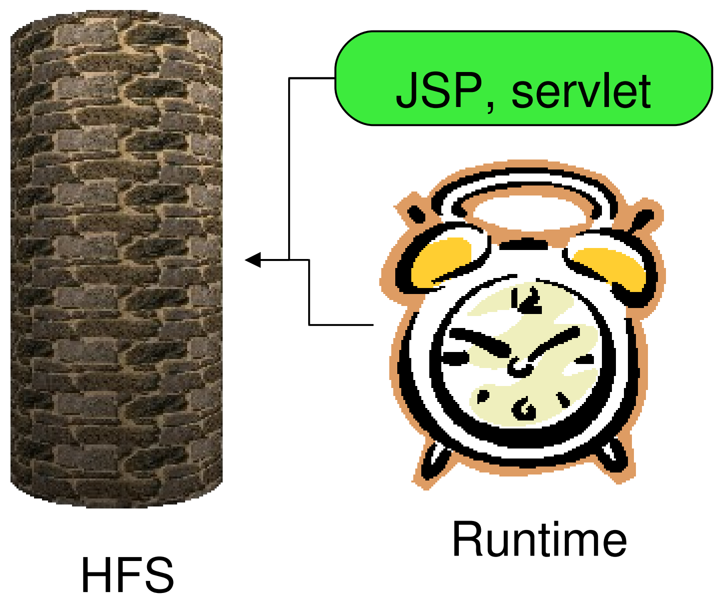

async beans
scheduler
startup beans
other PMEs
non-J2EE functionality cannot use JSR88
Use WAS administration for application deployment

Updating Your Application – v5

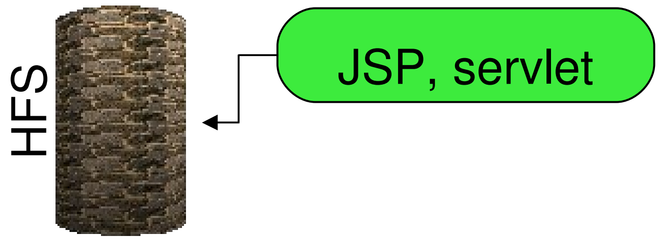
Update the entire application



Use dynamic reloading



Change file in HFS and restart



Not granular enough!

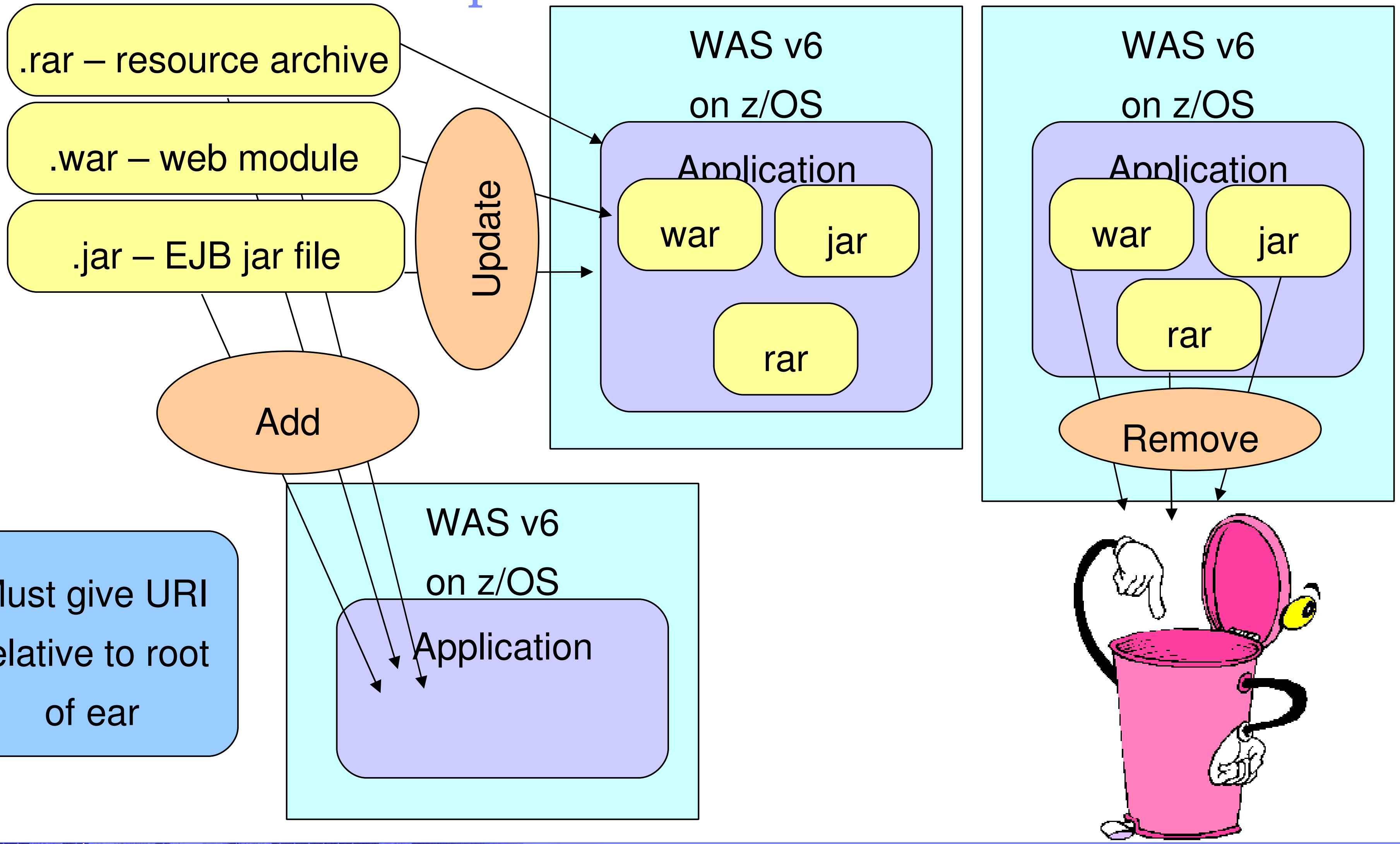
Updating Your App – v6 – Fine Grained Updates

Ways to update your application in WAS v6

- Full application
  - Same as in v5
- Individual modules
  - war, rar, jar
- Single file
  - JSP, HTML, images
- Partial application
  - Use a zip file

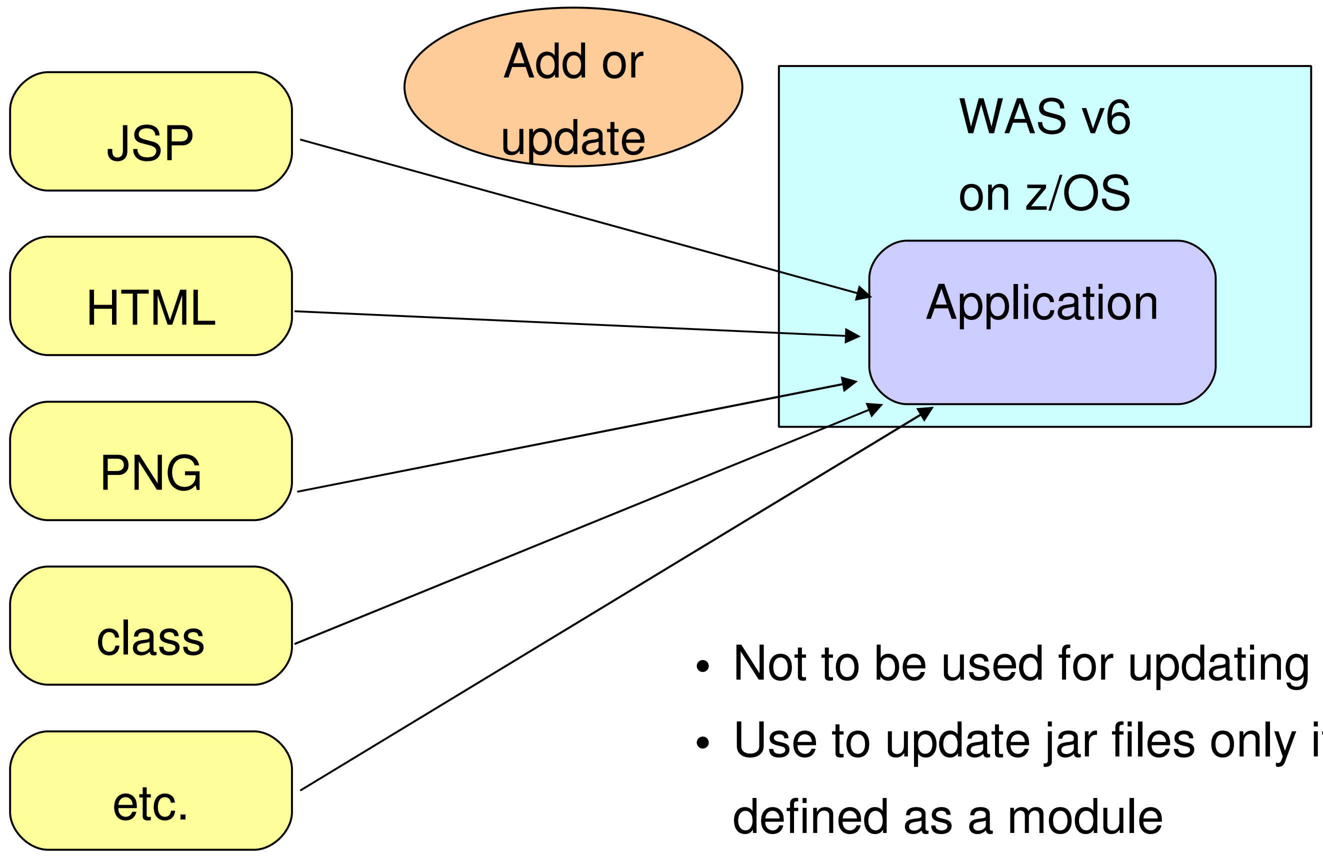
Details ahead...

Individual Module Update



Must give URI relative to root of ear

Updating Individual Application Files

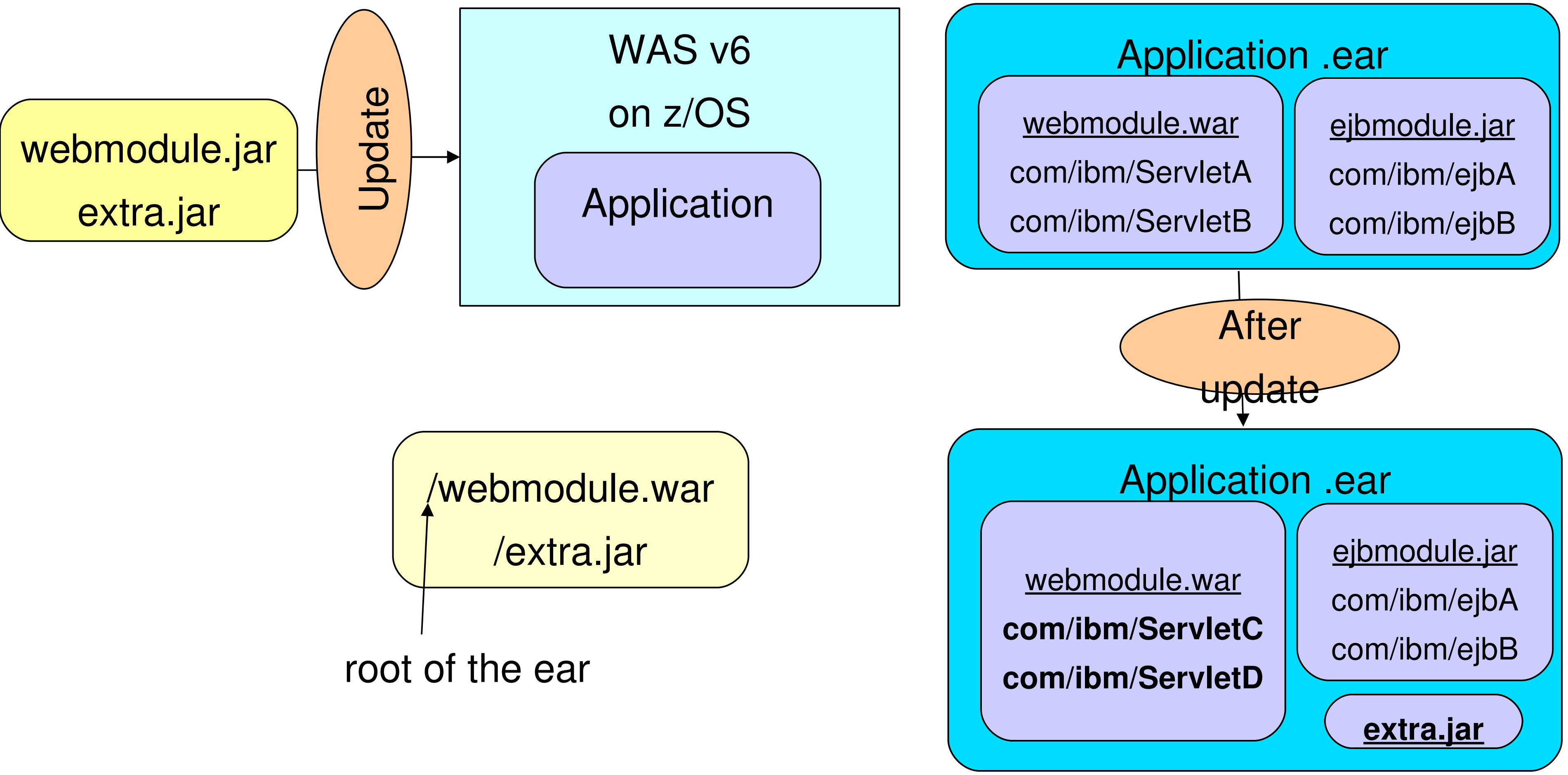


- Not to be used for updating war, rar, or ear
- Use to update jar files only if they are not defined as a module
- <u>Must give URI relative to the root of the ear</u>

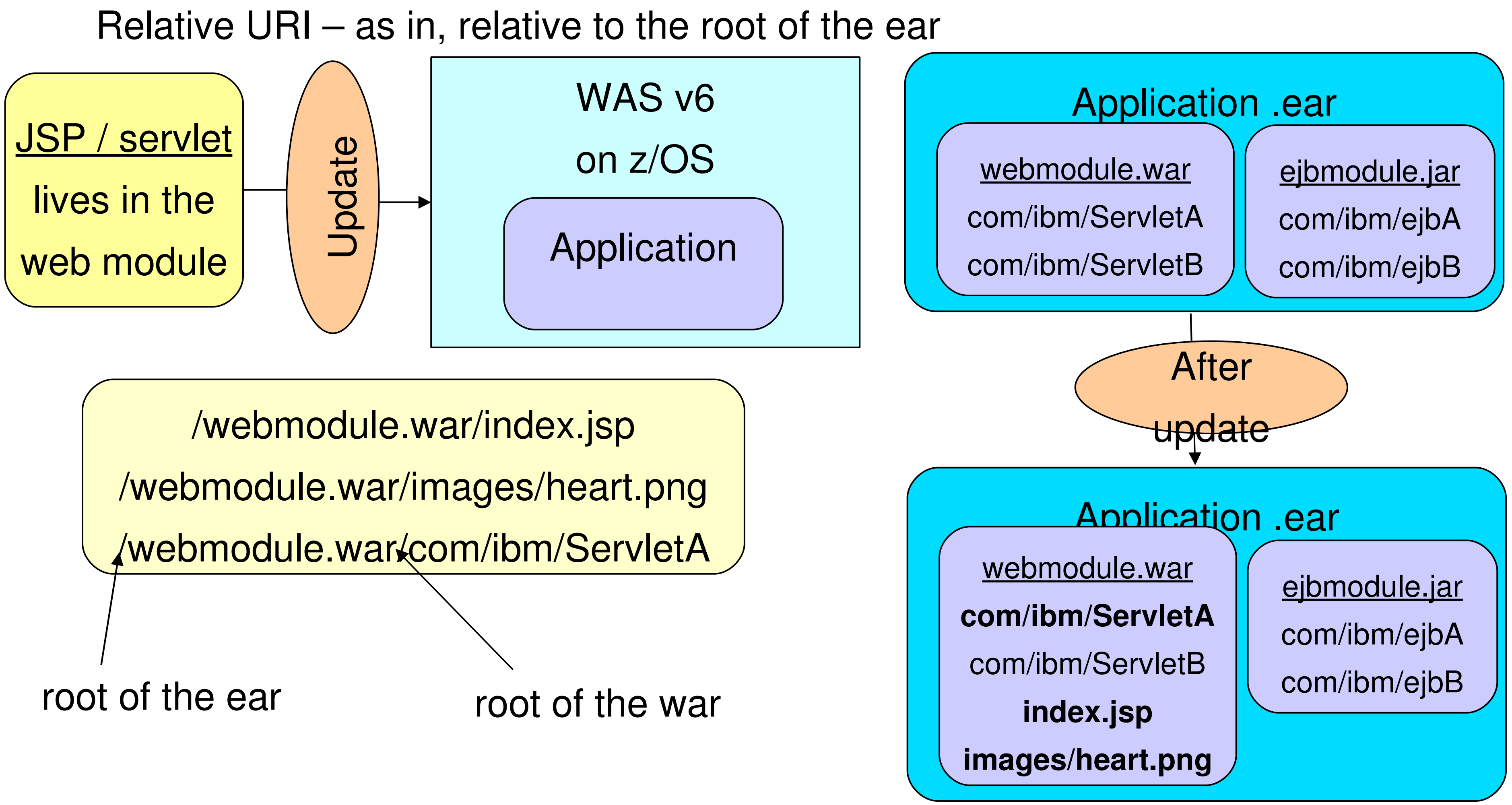
More on this later...

Relative URIs for Updating – Modules

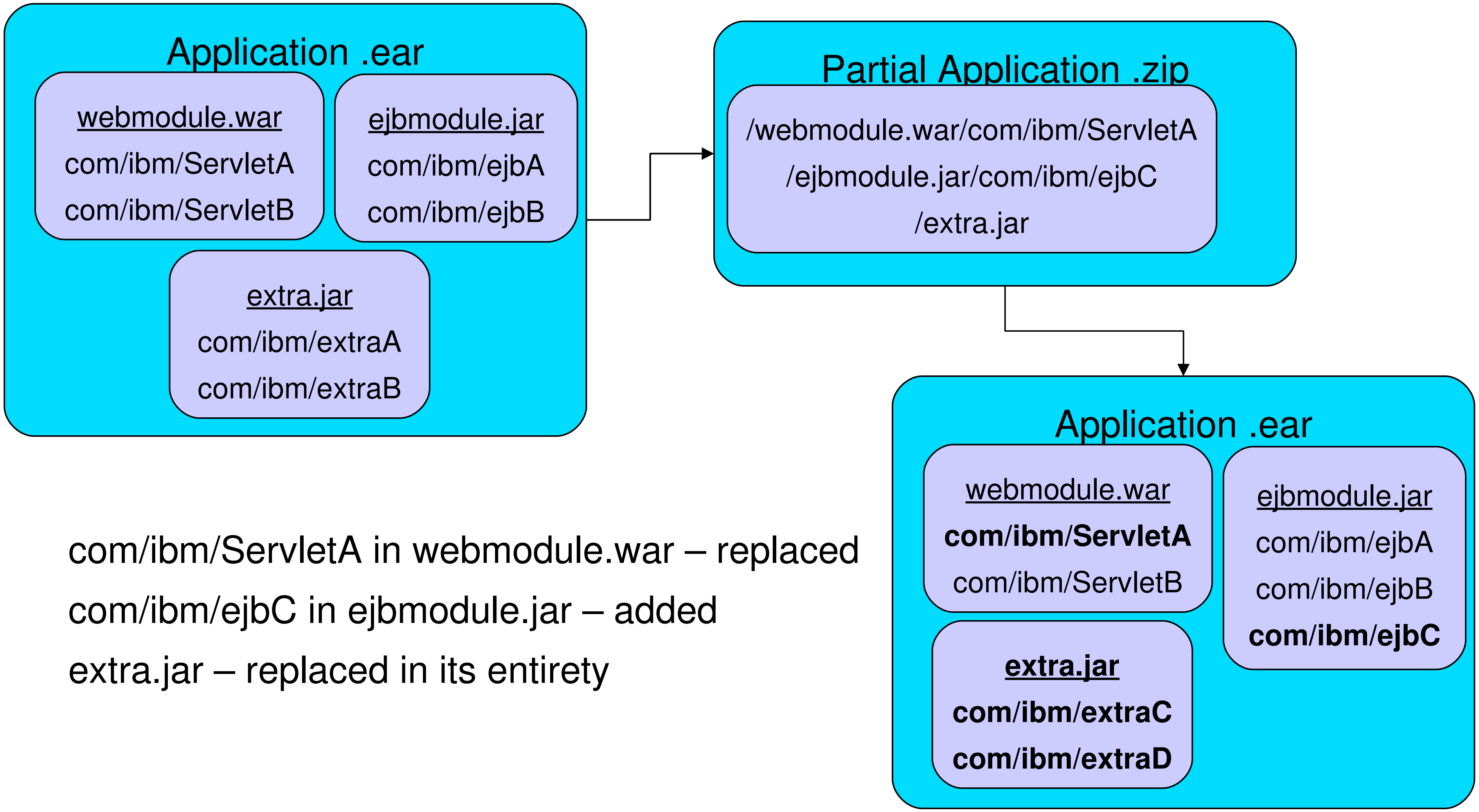
Relative URI – as in, relative to the root of the ear



Relative URIs for Updating – Individual Files



Update Using Partial Application



com/ibm/ServletA in webmodule.war – replaced

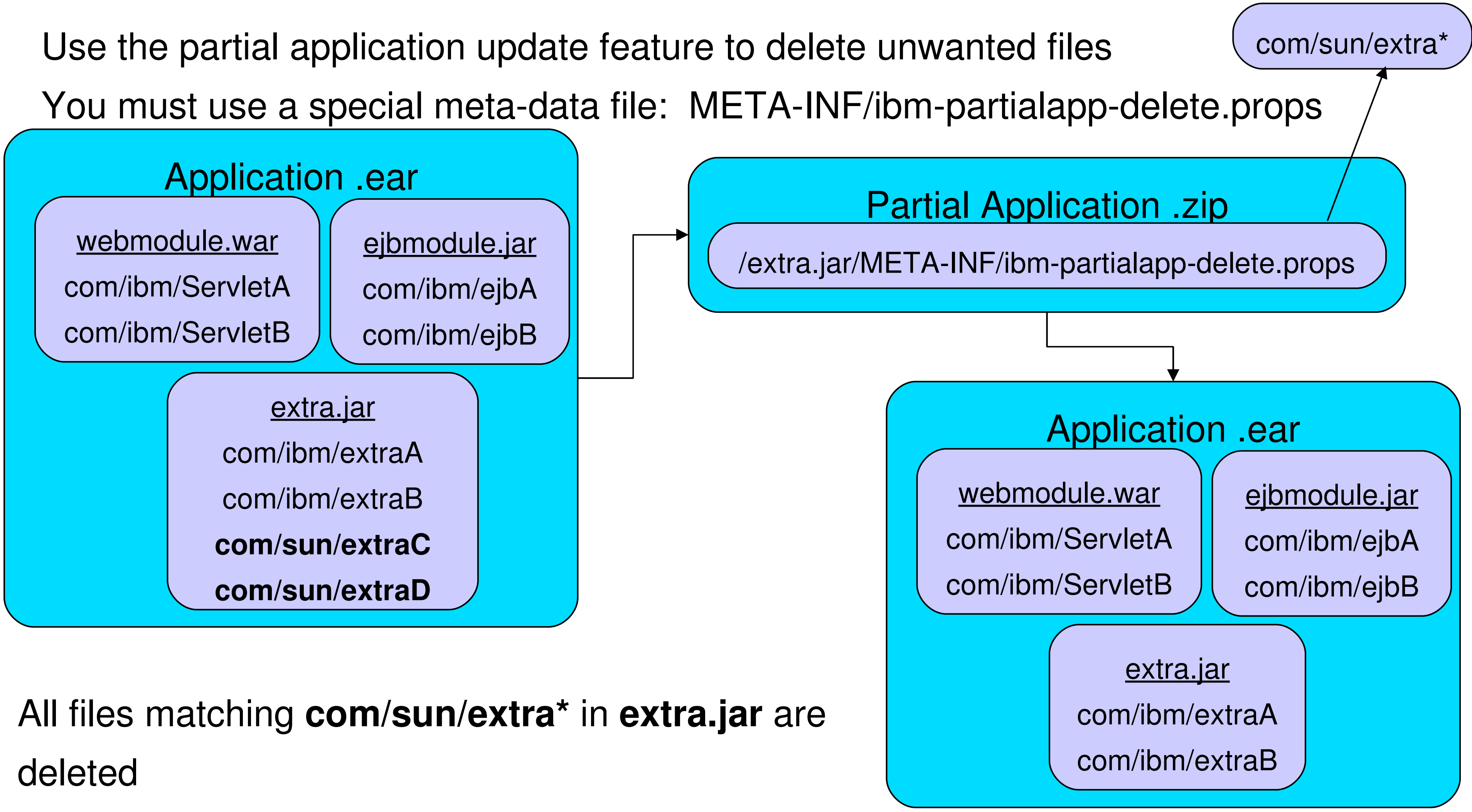
com/ibm/ejbC in ejbmodule.jar – added

extra.jar – replaced in its entirety

Deleting Files From Your Application

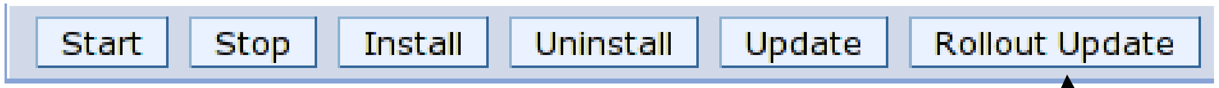
Use the partial application update feature to delete unwanted files

You must use a special meta-data file: META-INF/ibm-partialapp-delete.props



All files matching **com/sun/extra*** in **extra.jar** are deleted

Rollout Update



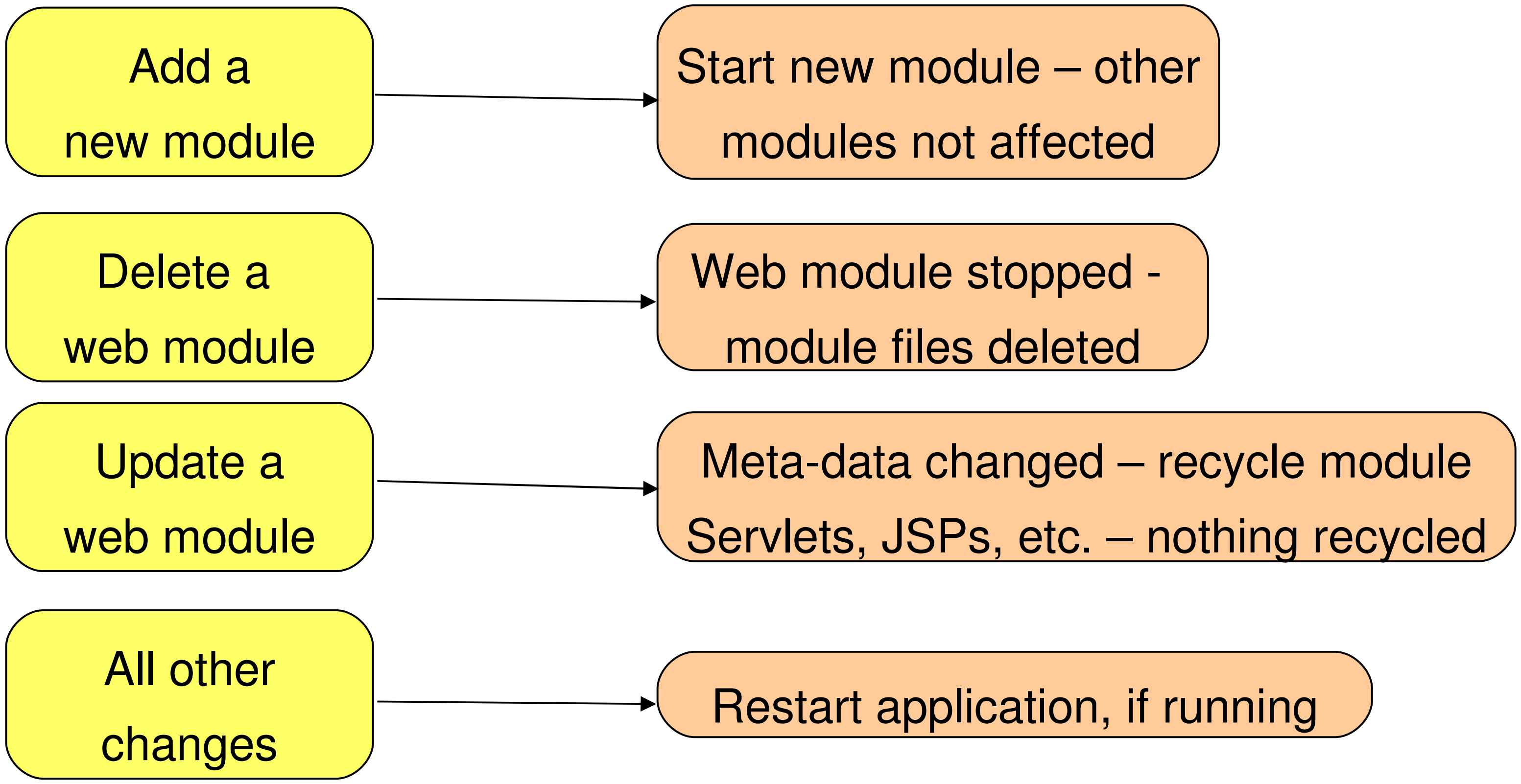
What is this new button?

- Sequentially updates an application installed on multiple clusters members
- Provides continuous available of the application
- Does the following for each cluster member in sequence
    1. Saves the updated application configuration
    2. Stops all of the cluster members on one node
    3. Updates the application on the node by synchronizing the configuration
    4. Restarts the stopped cluster members
    5. Repeat 1 - 4 for each node with cluster members

Application Restart Behavior

WAS v5 allowed only start / stop of entire application

WAS v6 allows module level start and stop operations


Add a new module
Start new module – other modules not affected
Delete a web module
Web module stopped - module files deleted
Update a web module
Meta-data changed – recycle module
Servlets, JSPs, etc. – nothing recycled
All other changes
Restart application, if running

Application Restart Behavior – Part 2

- Starting a module makes it accessible to clients
- Starting an application starts all its modules deployed to that server
    - Starting all application modules does not start the application
- Stopping an application makes it inaccessible to clients
    - Stopping all application modules does not stop the application
- EJB, web, and connector modules can be started at the module level
- Only web modules can be stopped
    - Only if the WAR classloader policy for the application is “module”
    - If the WAR classloader policy is “application,” the entire application needs to be recycled
- Users cannot start or stop the module; this only occurs as part of an update

WAR Classloader Policy

Controls the isolation of web modules

<u>Module</u>

each web module receives its own classloader whose parent is the application classloader (default)

Supports web module stopping

<u>Application</u>

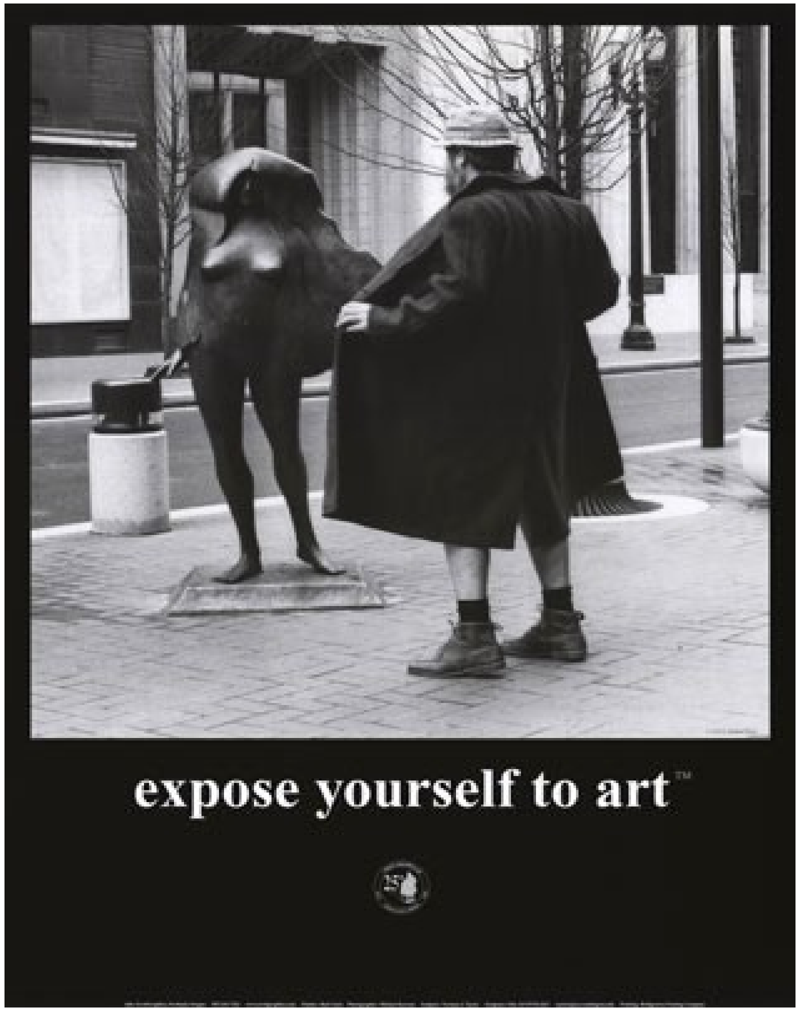
web module contents also loaded by the application classloader

In addition to EJB files, rar files, dependency files, and shared libraries

Application Update User Interface

Update API exposed via:

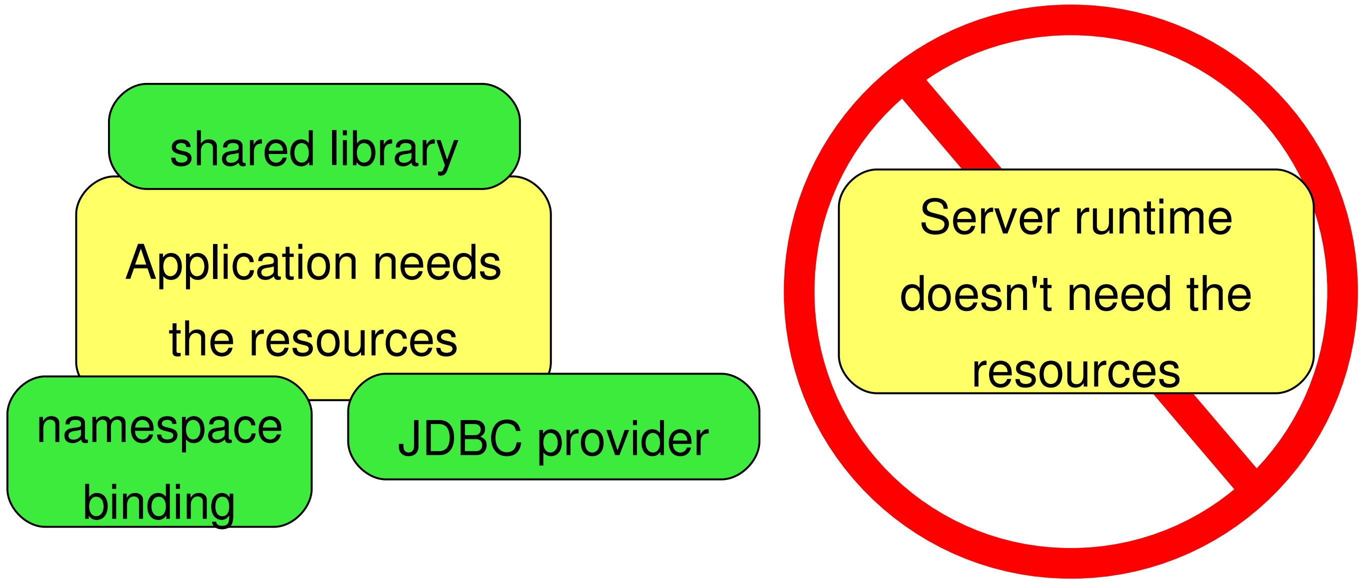
- Admin console
- wsadmin
    - $AdminApp update
- mbean interface
    - for creating your own custom mbean



Application Scoped Resources

•WAS v5 mechanism for storing resource definitions is topological (cell, node, etc.)

•If applications is moved, the resource definitions must be available in the new runtime


shared library
Application needs the resources
namespace binding
JDBC provider
Server runtime doesn't need the resources

•WAS v6 supports J2EE resource definitions in the application context of the config tree

•ASTK and RAD can be used to view / modify application scope resource

Enhanced EAR

<u>WAS v5</u>

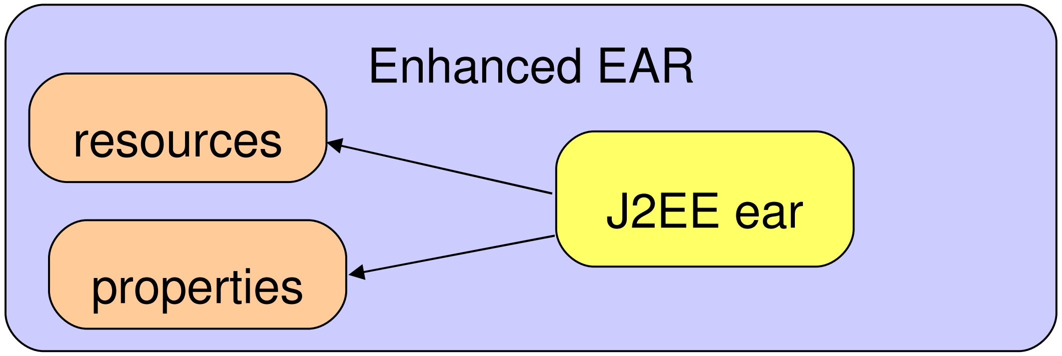
exporting an application means you lose classloader or shared library information

<u>WAS v6</u>

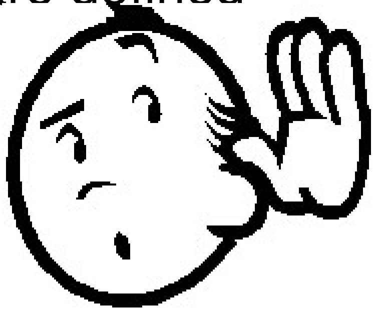
all applications exported as enhanced EAR

export --> update --> import is much easier

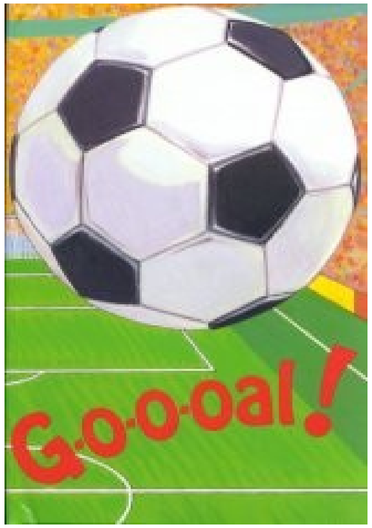
Enhanced EAR – Part 2



- An enhanced ear file contains configuration meta-data
- Users can define configuration for the application
    - virtual hosts, shared libraries, etc.
- Moving application from server to server is much easier
    - resources move with the application
- Resources created during deploy time are application scope
- Enhanced ear is detected during deploy and configurations are defined as specified by the config data
- RAD and ASTK support enhanced EAR import / export
- Enhanced EAR is not part of the J2EE spec



Why WebSphere Rapid Deploy (WRD)?



WRD has two goals

Simplify development

- Fewer artifacts to product and maintain
- Fewer concepts and technologies to understand

Simplify deployment

- Automated application deployment
- Reduced amount of information that must be collected by user to deploy
- Automated process for incremental changes

What is WRD?

WRD has two main concepts

Annotation-based programming

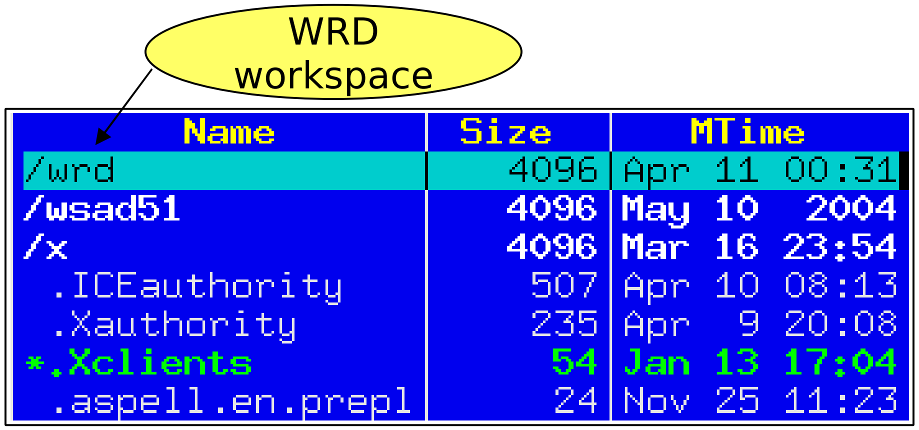
Deployment Automation

Used within RAD or ASTK

IBM
Rational. Software Development Platform
Version 6.0
Powered by Eclipse Technology
WebSphere. software
crystal reports.
Java

Licensed Material - Property of IBM Corp. (c) Copyright by IBM Corp. and other(s) 2000, 2004. All Rights Reserved. IBM, Rational, and WebSphere are trademarks of IBM Corp.; Crystal Reports is a registered trademark of Business Objects SA; Java and all Java-based marks and logos are trademarks or registered trademarks of Sun Microsystems, Inc.; and all terms are trademarks or registered trademarks in the United States, other countries, or both. Portions based on *Design Patterns: Elements of Reusable Object-Oriented Software*, by Erich Gamma, Richard Helm, Ralph Johnson and John Vlissides, Copyright (c) 1995 by Addison-Wesley Publishing Company, Inc. All rights reserved.

Headless install from a directory in the file system, known as the WRD workspace

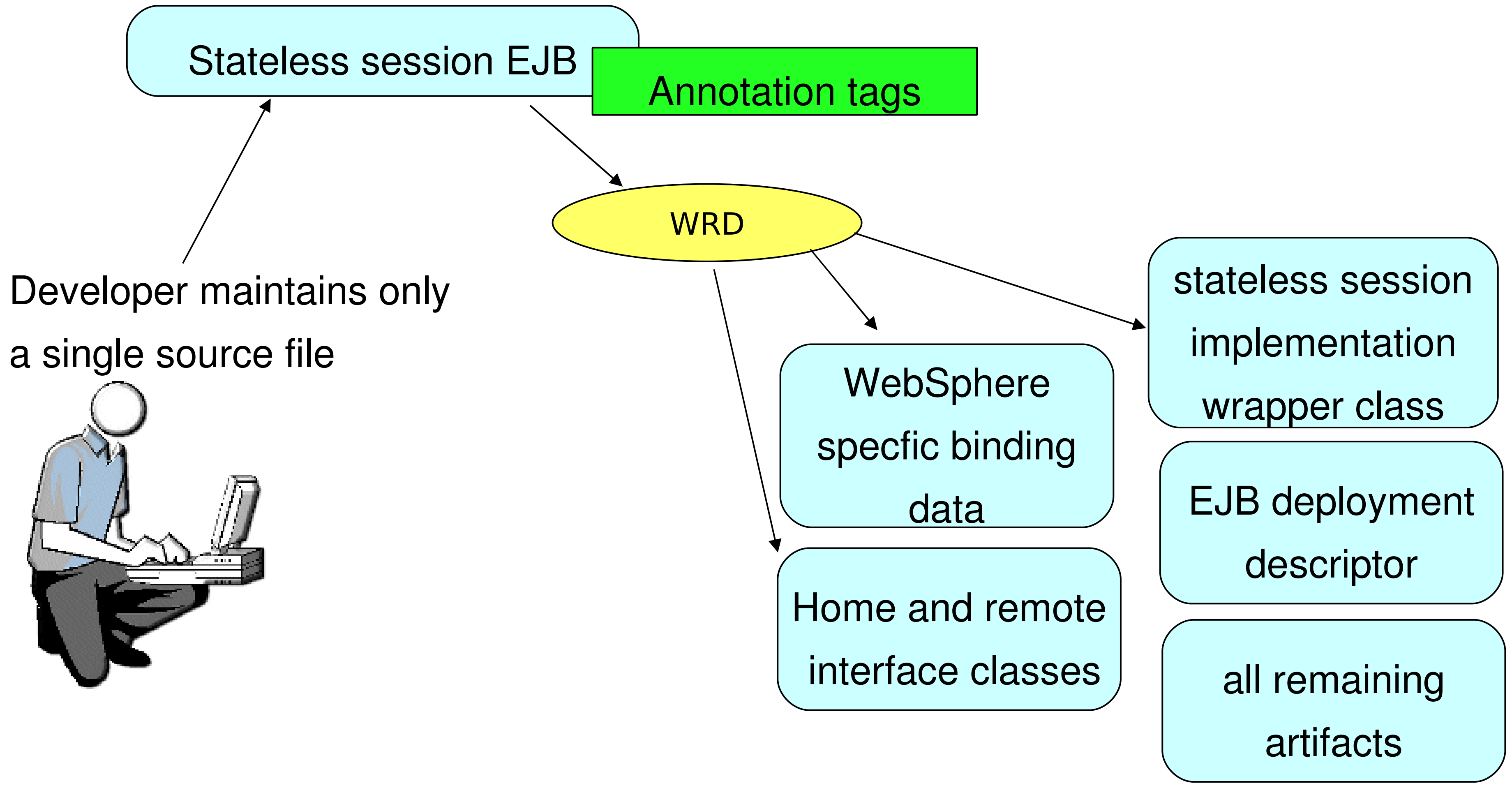


```
      Name            Size     MTime
/wrd                  4096 Apr 11 00:31
/wsad51               4096 May 10  2004
/x                    4096 Mar 16 23:54
 .ICEauthority         507 Apr 10 08:13
 .Xauthority           235 Apr  9 20:08
*.Xclients              54 Jan 13 17:04
 .aspell.en.prepl       24 Nov 25 11:23
```

Annotation-Based Programming

- Offers a set of tags that developer embeds into application source code
- WRD uses these tags to generate application artifacts necessary to execute the application
- Minimizes the number of artifacts the developer needs to create
- Developer only maintains a single artifact

Annotation-Based Programming – An Example


Stateless session EJB
Annotation tags
WRD
Developer maintains only a single source file
WebSphere specfic binding data
Home and remote interface classes
stateless session implementation wrapper class
EJB deployment descriptor
all remaining artifacts

Annotation Tags

- Annotations are used as Javadoc-style comments in the source
- Annotations can be included in the package, class, field, or method declarations
- WRD supports XDoclet syntax where it exists
  http://xdoclet.sourceforge.net/xdoclet/index.html
- Code-assist in RAD
- Entered using "@tag" in comment block
- WRD supports tags for:
  - •EJBs
  - •Servlets
  - •Java classes
  - •Web services

Annotation Tags – An Example

```
package com.creditcheckcorp.ejb;

import javax.ejb.SessionBean;
import javax.ejb.SessionContext;
import javax.ejb.CreateException;

/**
 * Bean implementation class for Session Bean: CheckCreditEJB
 *
 * @ejb.bean name="CheckCreditEJB" type="Stateless"
 *           jndi-name="ejb/com/creditcheckcorp/ejb/CheckCreditEJBHome"
 *           local-jndi-name="ejb/com/creditcheckcorp/ejb/CheckCreditEJBHome"
 *           view-type="both" transaction-type="Container"
 *
 * @ejb.home remote-class="com.creditcheckcorp.ejb.CheckCreditEJBHome"
 *           local-class="com.creditcheckcorp.ejb.CheckCreditEJBLocalHome"
 *
 * @ejb.interface remote-class="com.creditcheckcorp.ejb.CheckCreditEJB"
 *                local-class="com.creditcheckcorp.ejb.CheckCreditEJBLocal"
 *
 */

public class CheckCreditEJBBean implements SessionBean {
```

Types of Tags

Technology Tags

- Map directly to technologies in J2EE
- Mostly derived from XDoclet

Behavioral Tags

- Annotate a desired behavior or quality of service
- Does not indicate specific implementation
- WRD will determine appropriate implementation later

Bindings and Extensions Tags

- WebSphere-specific bindings
- Not part of the J2EE standard

Scope of Tags

Package

- Added to package comment
- Applicable to entire Java package, module, or application

Method

- Added to a method's comments
- Applicable to that particular method

Class

- Added to class comment
- Provides information about the Java type or interface as a whole

Field

- Added to a field's comments
- Applicable to that particular field

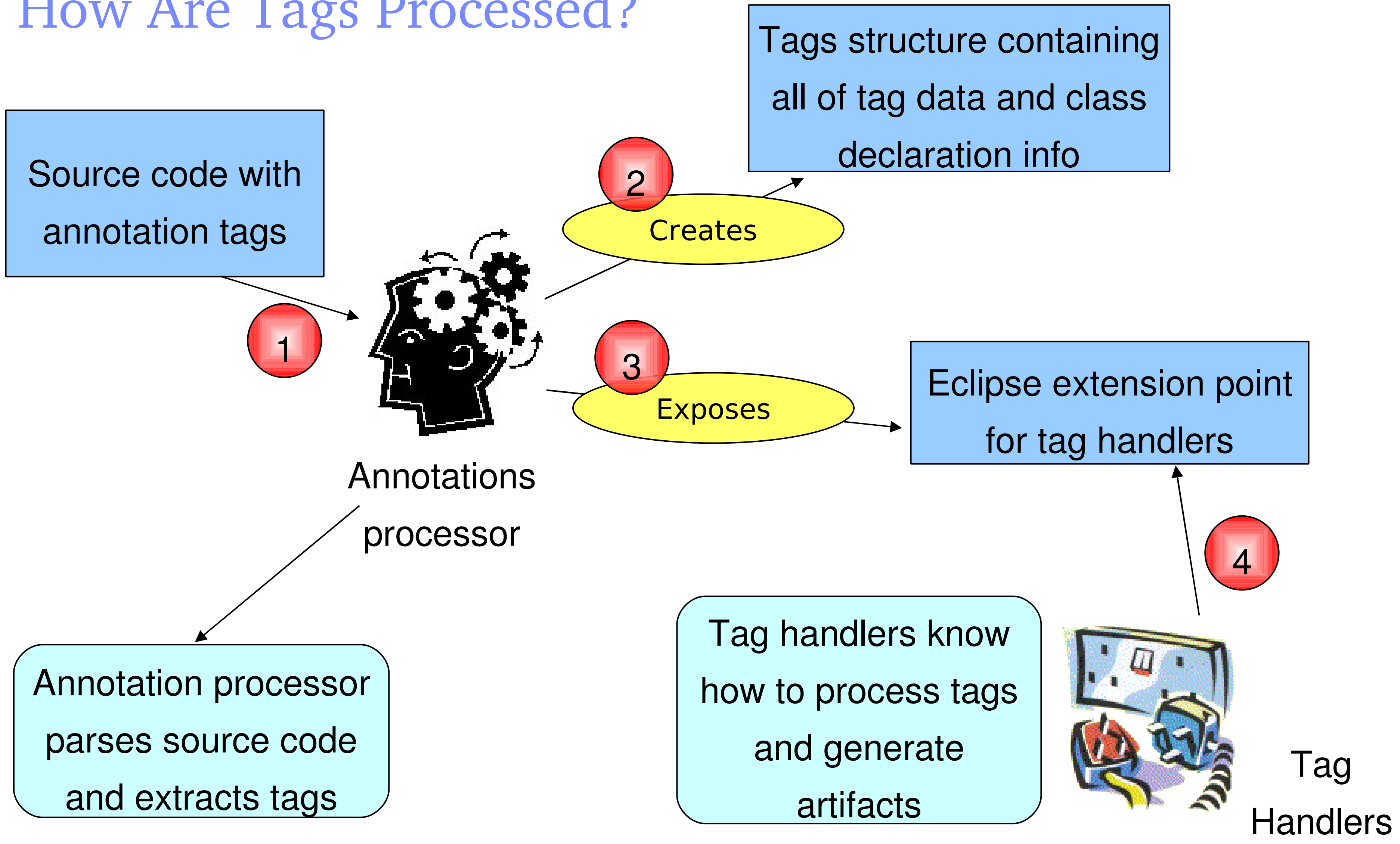
Annotation-Based Programming vs XDoclet

- XDoclet is a popular open source project
- “Attribute-oriented programming”
- Originally a tool for easing EJB development
- Processes annotations as part of the build process

- WRD adopts tag syntax used by XDoclet for J2EE 1.3
- Adoption of J2EE 1.4 tags when XDoclet 2 is released
- WRD contains proprietary tags for WebSphere-specific development

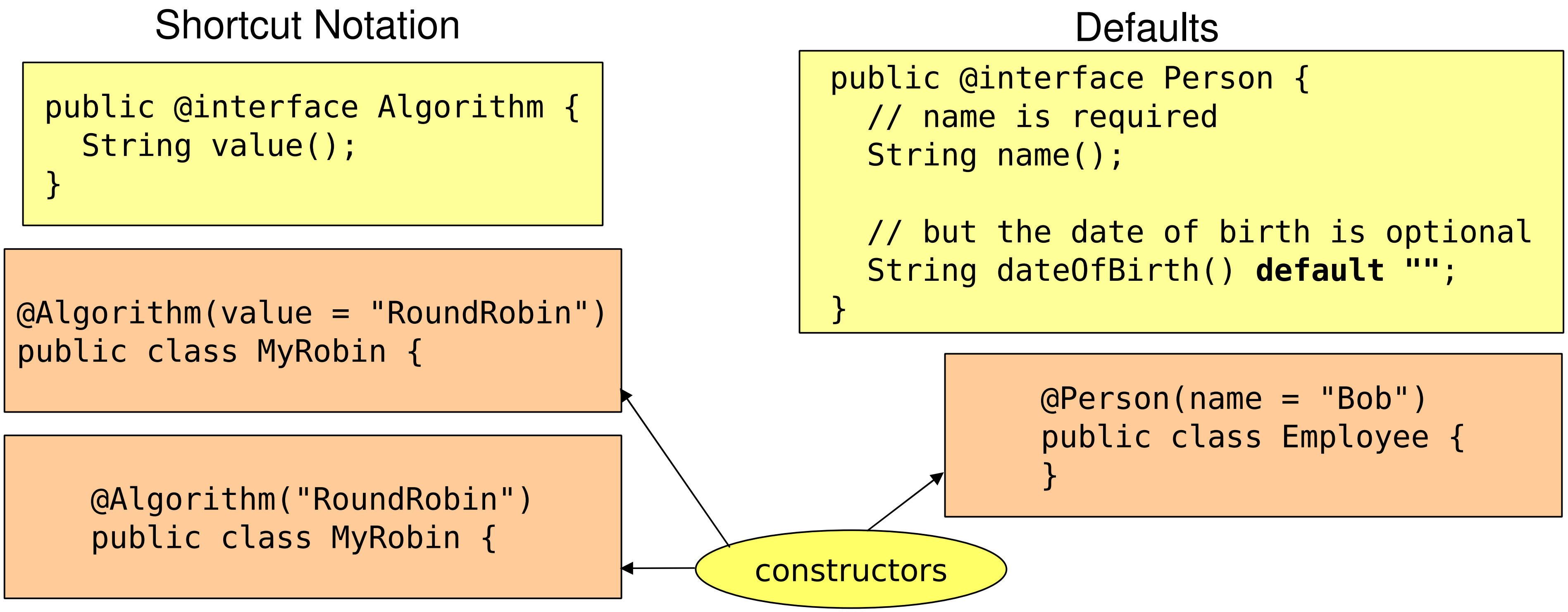
- WRD uses a different processing model
- WRD supports incremental, on-demand processing

How Are Tags Processed?


Source code with annotation tags
1
Annotations processor
2
Creates
Tags structure containing all of tag data and class declaration info
3
Exposes
Eclipse extension point for tag handlers
Annotation processor parses source code and extracts tags
Tag handlers know how to process tags and generate artifacts
4
Tag Handlers

What's In Store For Annotation-Based Programming?

- JSR 175 – meta-data facility for the Java programming language
- Adds meta-data tags into the Java language
- Standard set of tags for generating artifacts



Shortcut Notation

```
public @interface Algorithm {
  String value();
}
```

```
@Algorithm(value = "RoundRobin")
public class MyRobin {
```

```
    @Algorithm("RoundRobin")
    public class MyRobin {
```

Defaults

```
public @interface Person {
  // name is required
  String name();

  // but the date of birth is optional
  String dateOfBirth() default "";
}
```

```
    @Person(name = "Bob")
    public class Employee {
    }
```

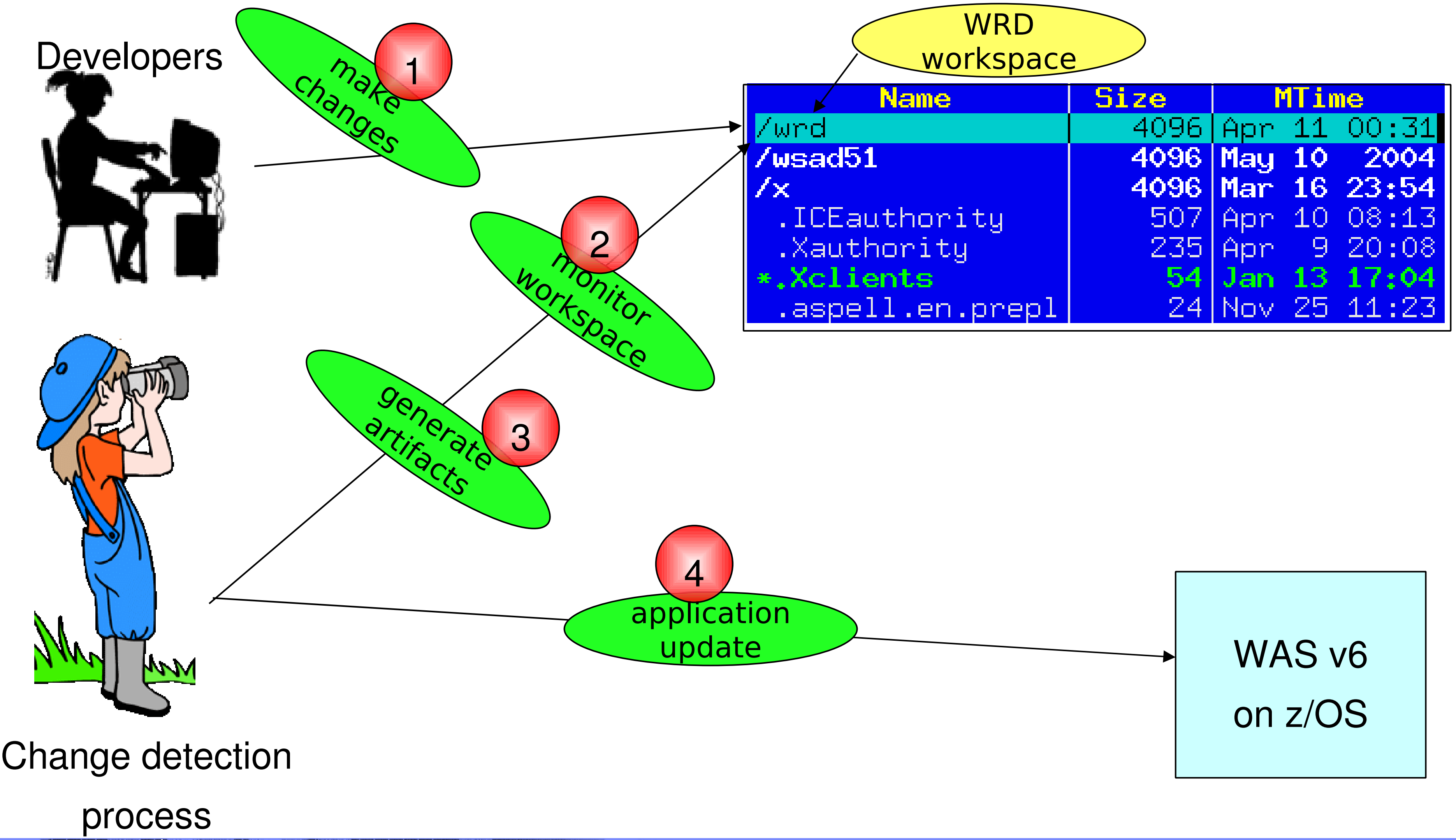
Deployment Automation

- Automatic deployment of applications onto local or remote WAS
- Free form application development
- “Hot directory” - used for file copy and “notepad” development
- Constructs a well-formed ear file from individual artifacts
- Support for fine grained application changes

WRD – Change Detection

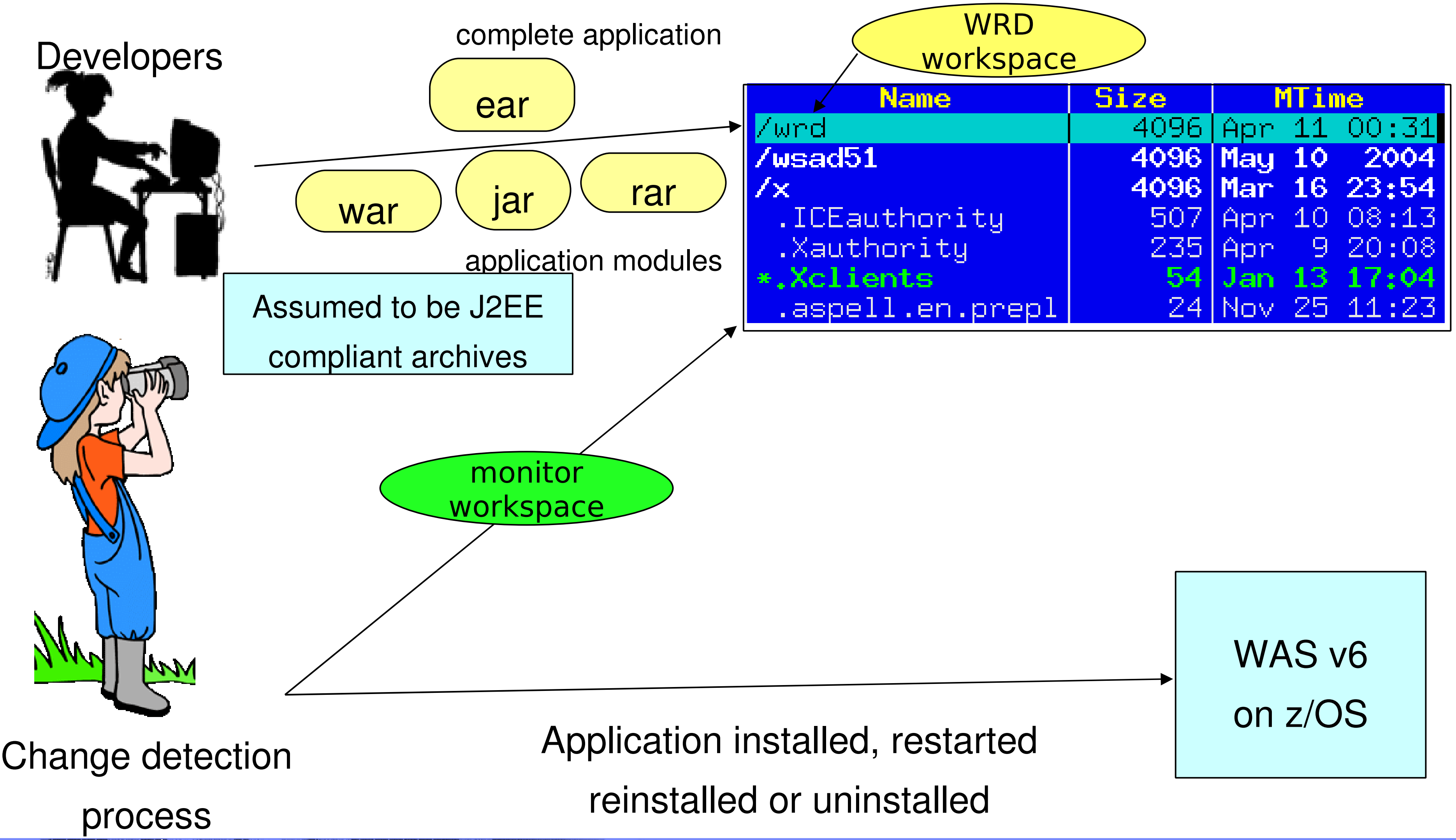
- Monitors file system for changes in WRD workspace
- Detection of change in application artifacts
    - Generates new application artifacts from existing artifacts
    - Deploys application changes to target server

WRD – Change Detection – Part 2



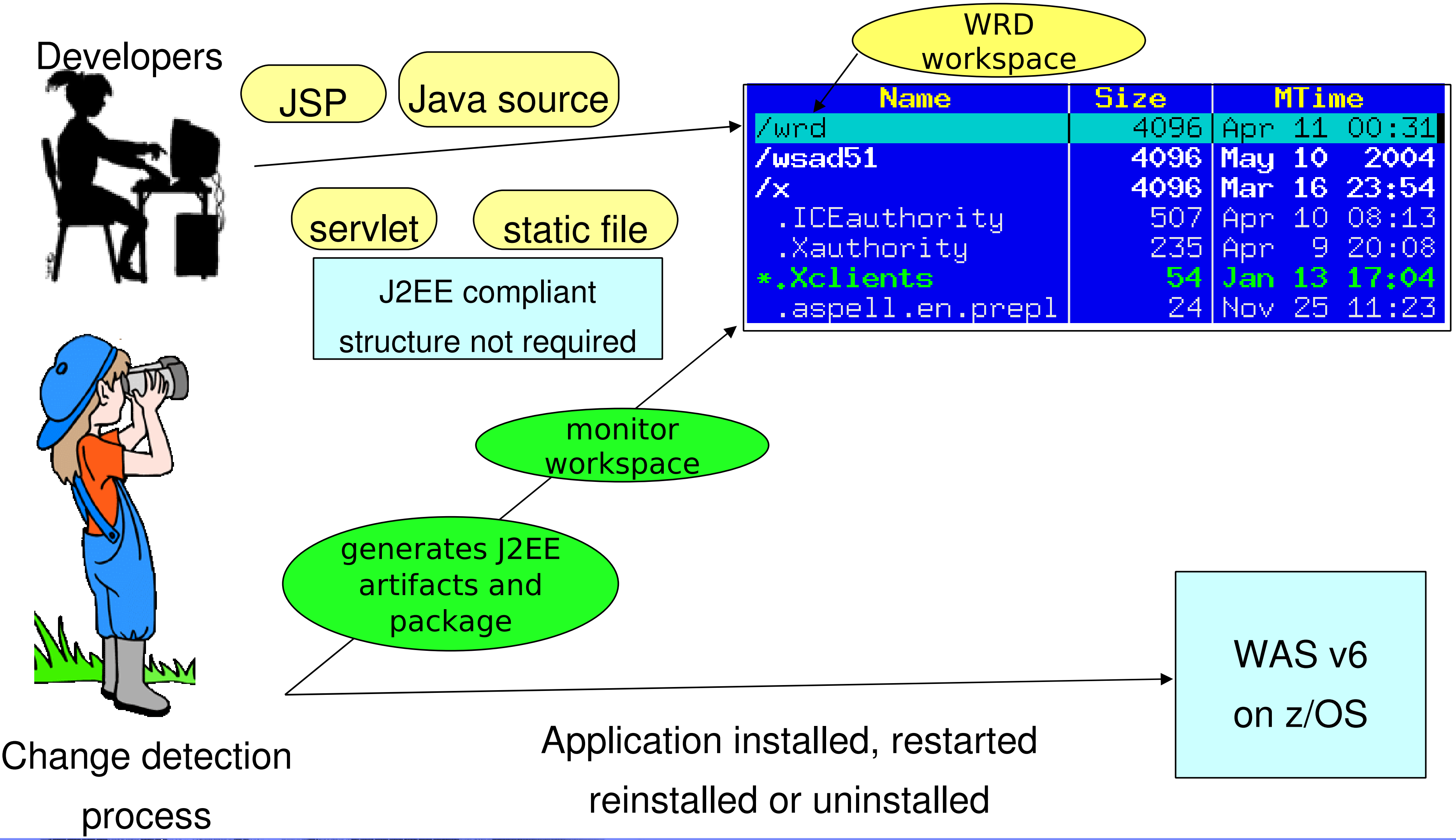
```
Name                  Size  MTime
/wrd                  4096  Apr 11 00:31
/wsad51               4096  May 10  2004
/x                    4096  Mar 16 23:54
  .ICEauthority        507  Apr 10 08:13
  .Xauthority          235  Apr  9 20:08
*.Xclients              54  Jan 13 17:04
  .aspell.en.prepl      24  Nov 25 11:23
```

WRD Styles - autoappinstall



```
Name                    Size  MTime
/wrd                    4096  Apr 11 00:31
/wsad51                 4096  May 10  2004
/x                      4096  Mar 16 23:54
  .ICEauthority          507  Apr 10 08:13
  .Xauthority            235  Apr  9 20:08
*.Xclients                54  Jan 13 17:04
  .aspell.en.prepl        24  Nov 25 11:23
```

WRD Styles – freeform



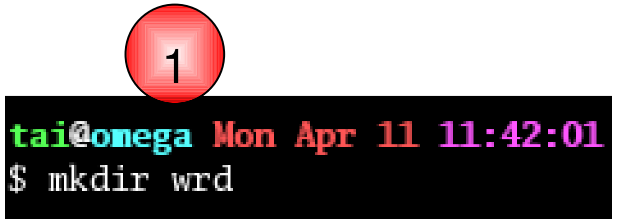
```
Name                   Size     MTime
/wrd                   4096     Apr 11 00:31
/wsad51                4096     May 10  2004
/x                     4096     Mar 16 23:54
 .ICEauthority          507     Apr 10 08:13
 .Xauthority            235     Apr  9 20:08
*.Xclients               54     Jan 13 17:04
 .aspell.en.prepl        24     Nov 25 11:23
```

Setting Up Deployment Automation

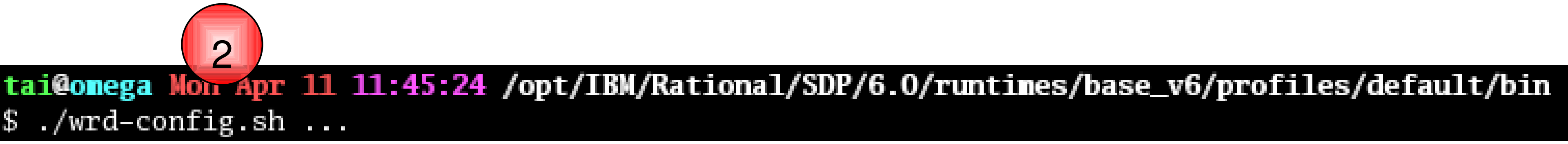
- Uses Eclipse framework
- No GUI
- Uses a set of command-line scripts
  - Found under <profile_home>/bin

Three Quick Steps to Using WRD

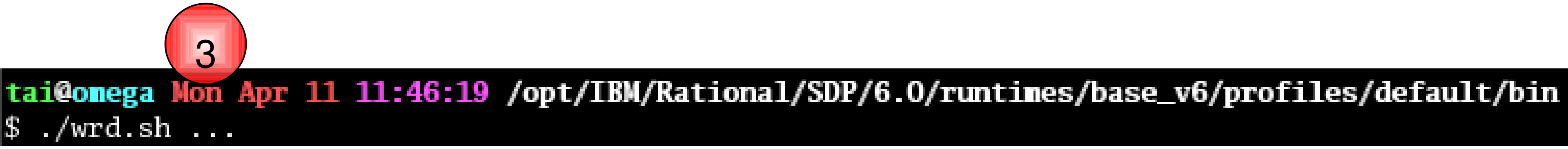
1. Configure WRD workspace
2. Configure WRD project (name, style)
3. Enable WRD monitoring process


1

```
tai@omega Mon Apr 11 11:42:01
$ mkdir wrd
```


2

```
tai@omega Mon Apr 11 11:45:24 /opt/IBM/Rational/SDP/6.0/runtimes/base_v6/profiles/default/bin
$ ./wrd-config.sh ...
```


3

```
tai@omega Mon Apr 11 11:46:19 /opt/IBM/Rational/SDP/6.0/runtimes/base_v6/profiles/default/bin
$ ./wrd.sh ...
```

Configuring a WRD Workspace

UNIX:
wrd-config.sh

Windows:
wrd-config.bat

- Use the wrd-config.(bat|sh) script at the command line
- Found under <profile_home/bin directory

mandatory

```
wrd-config.bat -project “projectname” -style “freeform|autoappinstall” [optional parameters]
```

-rebuild – cleans and rebuilds workspace
-configure – interactive session
-runtime “was51|was60” – specify target runtime
-usage – displays help

see next slide...

WRD Configuration Parameters

wrd-config.(bat|sh) optional parameters

-rebuild – cleans and rebuilds workspace

-configure – interactive session

-runtime “was51|was60” – specify target runtime

-runtimePath – directory where target runtime is installed

-j2eeVersion “1.3|1.4” – specify target J2EE spec level

-configPath – path to XML file where workspace config will be persisted

-configData – path to XML file containing workspace config data

-listStyles – available deployment styles and descriptions

-listServers – available runtime server targets

-properties – properties for given deployment project

-buildMode – disable all console output

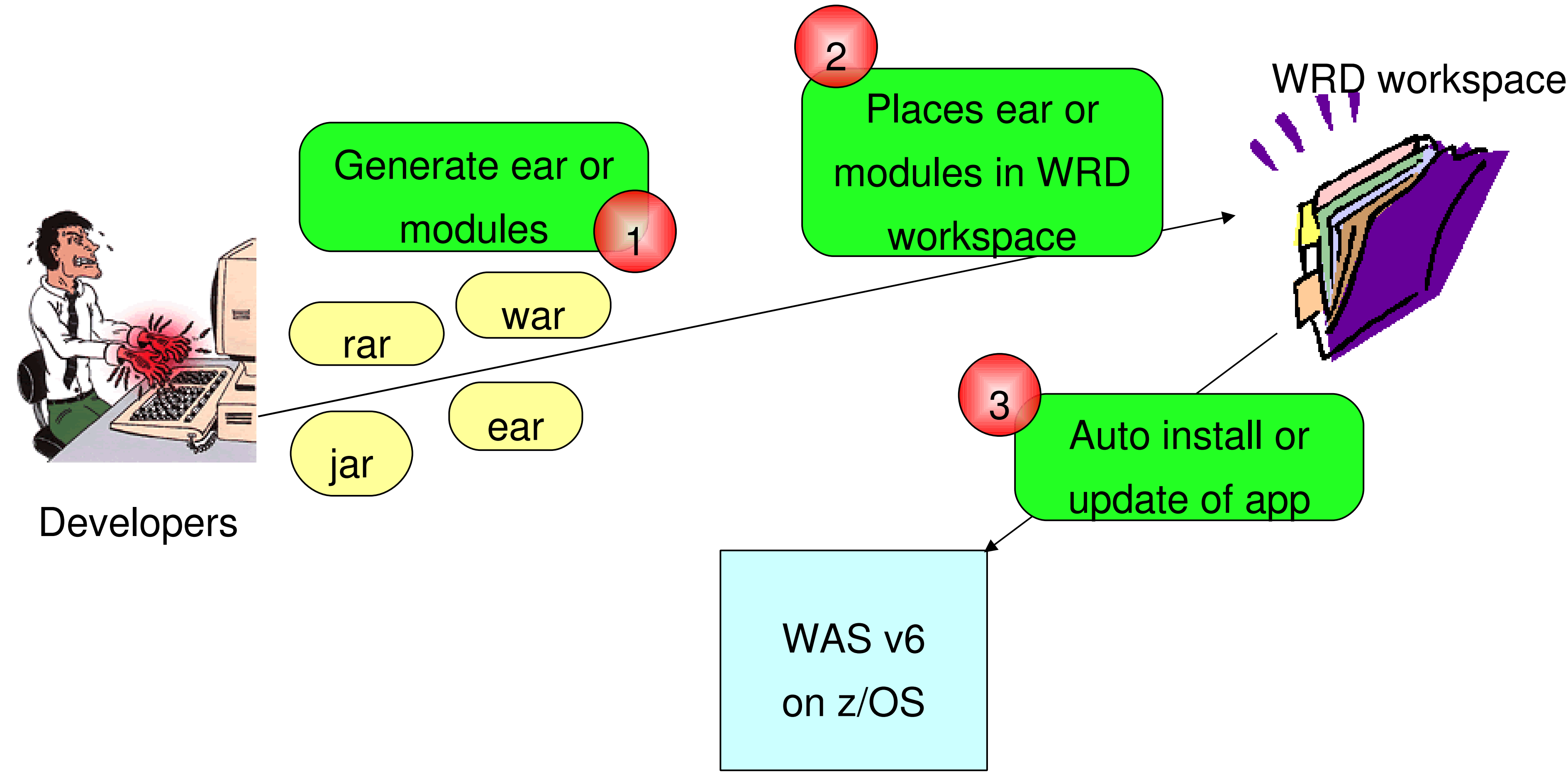
-usage – displays help

Running WRD

- Enable change detection monitor
- Found at <profile_home>/bin/wrd.(bat|sh)
- Additional parameters
  - -monitor – enable console output
  - -batch – build WRD workspace and shutdown

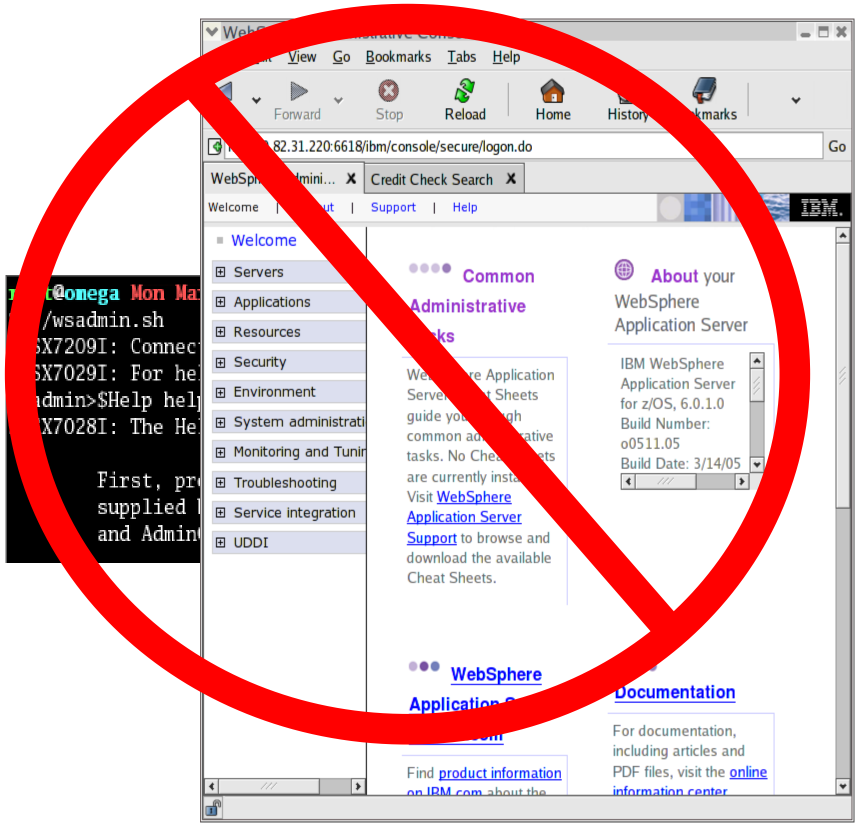
When Do I Use autoappinstall Style?

Useful for creating / updating complete applications or modules



What Does This Mean for Us?

No more admin console or wsadmin to install / update / reinstall!



```
t@omega Mon Ma
/wsadmin.sh
SX7209I: Connec
SX7029I: For he
dmin>$Help hel
X7028I: The He

      First, pr
      supplied
      and Admin
```

Common Administrative ks

We re Application Server t Sheets guide yo gh common ad ative tasks. No Che ts are currently insta Visit WebSphere Application Server Support to browse and download the available Cheat Sheets.

About your WebSphere Application Server

IBM WebSphere Application Server for z/OS, 6.0.1.0
Build Number: o0511.05
Build Date: 3/14/05

WebSphere Application

Find product information on IBM.com about the

Documentation

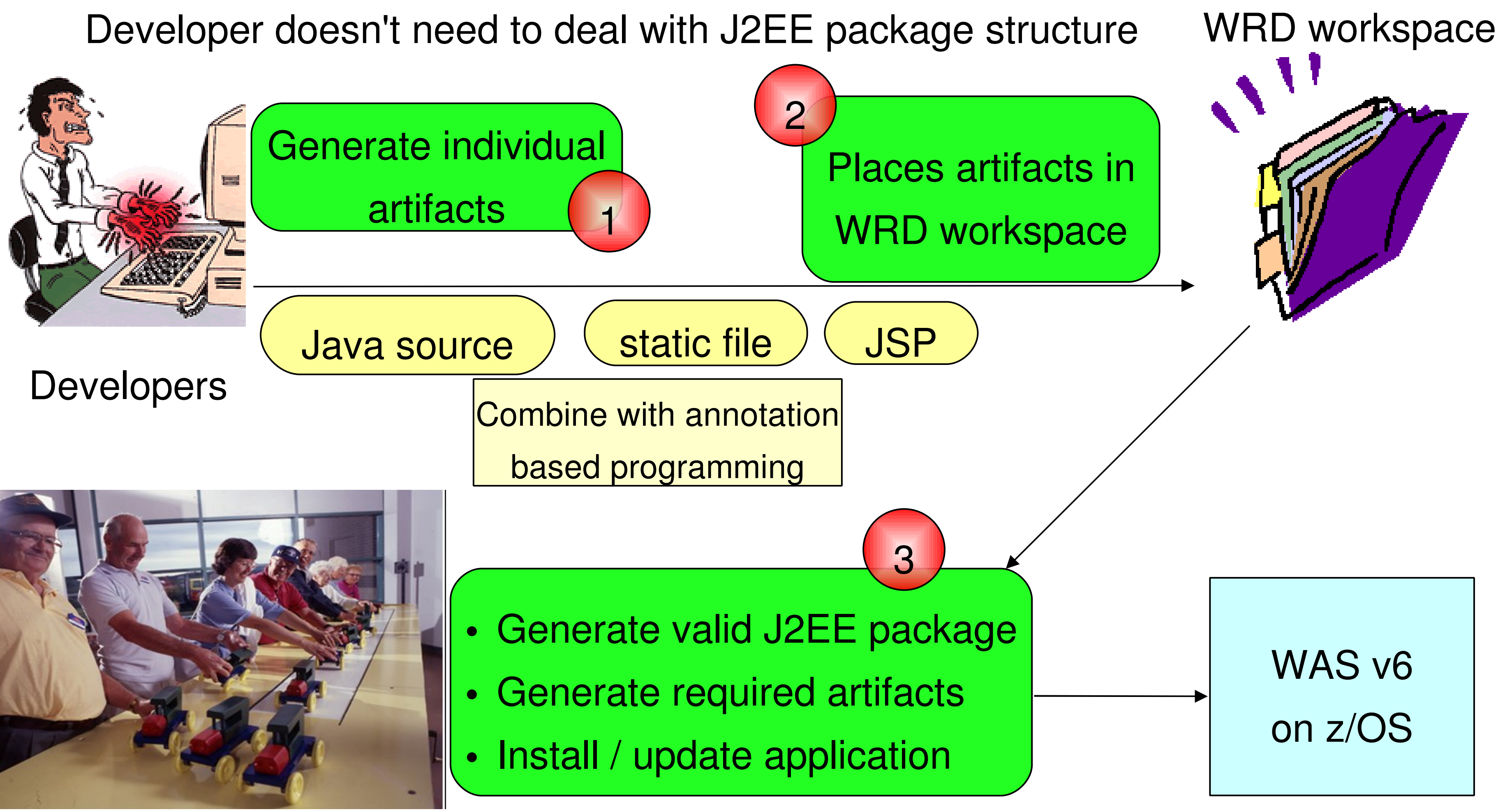
For documentation, including articles and PDF files, visit the online information center

Simplified deployment!

When Do I Use freeform Style?

Useful for creating individual artifacts

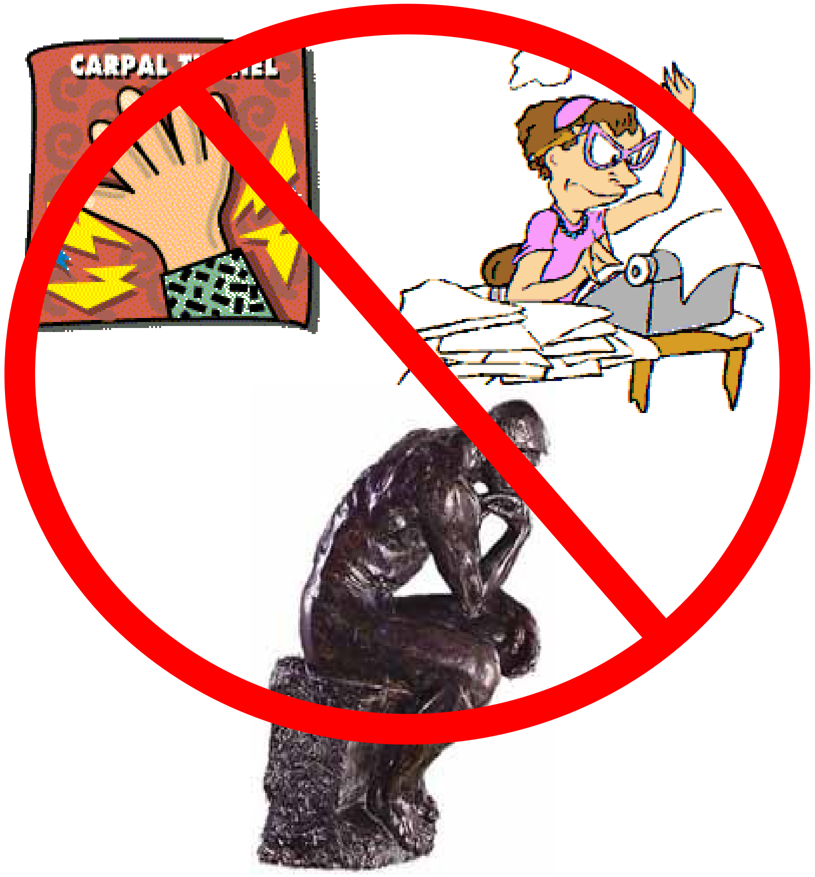
Developer doesn't need to deal with J2EE package structure


WRD workspace
Developers
1 Generate individual artifacts
2 Places artifacts in WRD workspace
Java source
static file
JSP
Combine with annotation based programming
3
• Generate valid J2EE package
• Generate required artifacts
• Install / update application
WAS v6 on z/OS

What Does This Mean For Us?

Less development

No dealing with complexity of J2EE package structure



Simplified development!

Questions?