

Advanced Weather Interactive Processing System II (AWIPS II)

AWIPS Development Environment (ADE)
and the
Common AWIPS Visualization
Environment
(CAVE)

Module 17: CAVE Updates for TO10

February 18, 2009

AWP.TRG.SWCTR/TO10.ADE/CAVE-17.00

This document includes data that shall not be duplicated, used, or disclosed – in whole or in part – outside the Government for any purpose other than to the extent provided in contract DG133W-05-CQ-1067. However, the Government shall have the right to duplicate, use, or disclose the data to the extent provided in the contract. This restriction does not limit the Government's right to use information contained in this data if it is obtained from another source without restriction. The data subject to the restriction are contained in all sheets.



Objective

- Understand the modifications to the CAVE architecture that were implemented in AWIPS II TO10



Topics

- Describe CAVE platform updates
- Summarize additional capabilities added to CAVE
- Describe CAVE baseline reorganization
- Describe CAVE serialization improvements
- Describe modified error handling capability



Platform Updates



Platform Updates: Eclipse

- Update:
 - Eclipse has been updated to Version 3.4.1, built 9/11/2008
- Rationale:
 - Latest Version available at the appropriate time in the TO, contains latest bug fixes and enhancements
 - Used for CAVE & EDEX development and builds
- Impacts:
 - Minimal changes required
- Install:
 - packaged with AWIPS II Installers
- Important:
 - If you are doing CAVE development, you need to use the Eclipse that is packaged in the ADE Installer



Platform Updates: Other Packages

- Other software has been updated as needed to be compatible with the platform
 - Specific version information is available in the AWIPS II SVD document (included on the install media)



Questions?



Additional AWIPS I Functionality Added



Additional AWIPS I Functionality Added

- Additional functionality has been added to CAVE
 - Display of additional products
 - Continued work on the Volume Browser and derived parameters
 - Additional GFE functionality and hazards generation
 - Additional WarnGen functionality
- Guardian has been added; initial implementation
 - More on this later



Questions?



CAVE Baseline Reorganization



CAVE Baseline Reorganization

- There is a continuing effort to reorganize the AWIPS II code baseline
 - Most of the effort in TO 10 was expended in the EDEX baseline
- Only one copy of Eclipse is in the baseline
 - Located in the *uframe-eclipse* project
 - Both CAVE and EDEX use this Eclipse for builds
 - Developers should use this version as their IDE
- Most FOSS/COTS components are housed in separate projects
 - For example, the Apache Camel packages are in the `org.apache.camel` project in the code base line
 - Both CAVE and EDEX can be build using only the projects in the baseline

Note: all AWIPS II code and support components are now packaged as Eclipse (OSGi compliant) plug-ins!



CAVE Baseline Reorganization (cont'd)

New CAVE Projects

- Some CAVE projects have been deleted from the baseline
 - For example, the com.raytheon.edex project no longer exists
- There are a number of new projects in CAVE
 - Most represent added AWIPS I functionality
- Some of the projects have “uf” as part of the project name
 - Generally, this is intended to identify the project as being “core” capability
 - This allows for better control (at Raytheon) of who is authorized to modify the project; we are attempting to keep a tighter rein on the core code.



CAVE Baseline Reorganization (cont'd)

- All AWIPS II components are now OSGi/Eclipse plug-ins
 - Provides a consistent build/deploy model
 - Provides improved work flow for developers, especially on the EDEX side
- CAVE builds leverage the reorganization of the EDEX code base to simplify CAVE coding
 - The com.raytheon.edex project has been eliminated
 - Used to contain “client” jars from selected EDEX components
 - CAVE components are able to reference EDEX components directly
 - CAVE build automatically includes necessary EDEX components
 - CAVE developers do need the EDEX baseline for development work



Questions?



Data Serialization



Data Serialization

JAXB/Thrift Serialization

- CAVE has generally moved to use JAXB/Thrift for serialization
 - All shared (CAVE & EDEX) code uses JAXB/Thrift exclusively
 - See the section on serialization in Module 16 (EDEX) for details on serialization refactor and JAXB/Thrift

JiBX Serialization

- CAVE continues to use JiBX internally for serialization to XML
 - Used primarily for product bundles
- JiBX scheduled for elimination from CAVE in TO 11
 - New code should avoid using JiBX if at all possible



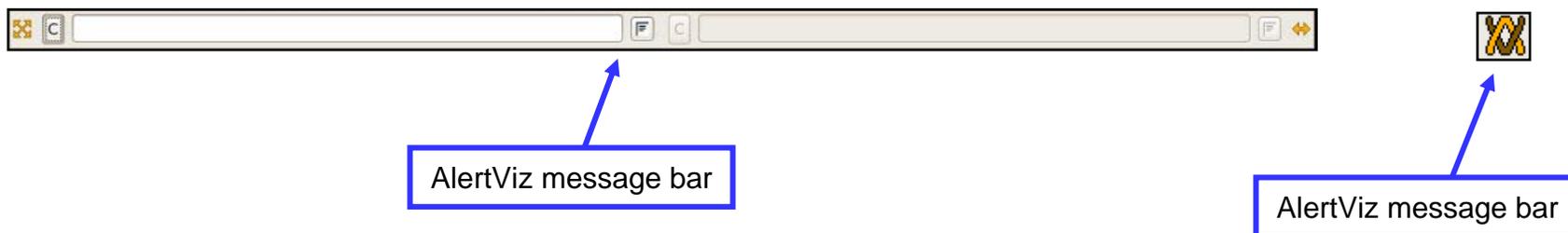
Questions?



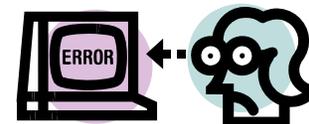
Guardian (AlertViz) – and Error Handling



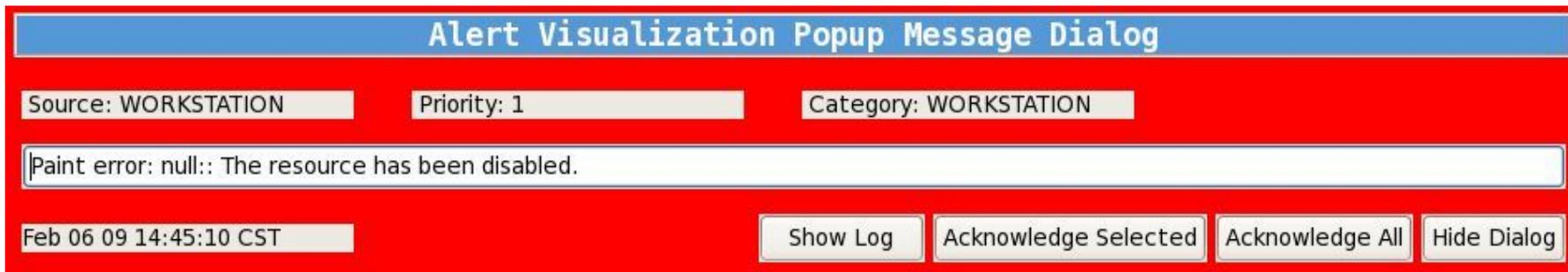
Guardian (AlertViz) – and Error Handling



- TO 10 adds AlertViz to CAVE
 - General alerting capability
- In normal operation, Guardian to main components
 - A floating message bar (above left)
 - Used to manage messages
 - A desktop toolbar menu (above right)
 - Used to configure AlertViz



Guardian (AlertViz) – and Error Handling (cont'd)

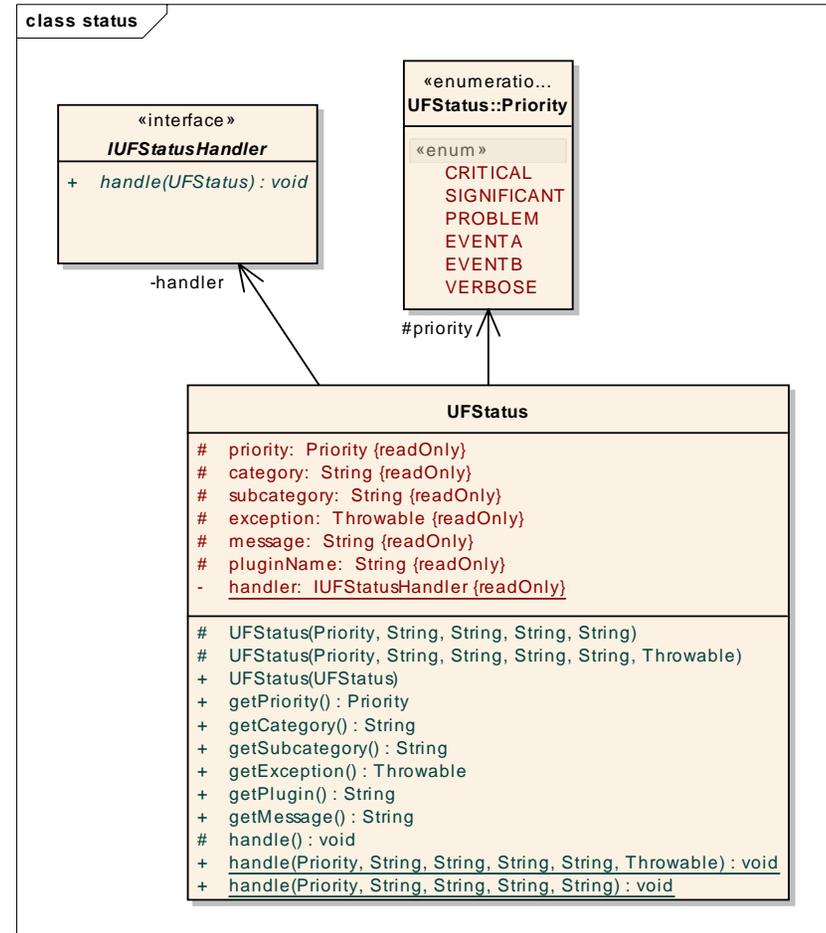


- Messages received by AlertViz can pop up for immediate attention/action
 - Shown here is an error generated attempting to load a product
- User acknowledges messages, etc.



Guardian (AlertViz) – and Error Handling (cont'd)

- For Guardian to work optimally, all logging performed within CAVE needs to use the UF Status message handling facility
 - Provided by *UFStatus*
 - Located in the *com.raytheon.uf.common.status* component in CAVE
- Client code calls one of the static *handle(...)* methods
- Message priorities are provided by the *UFStatus*. Priority enumeration



Guardian (AlertViz) – and Error Handling (cont'd)

```
try {
    baseColormapDir = PathManagerFactory.getPathManager().getFile(
        edexStaticBase, "colormaps").getCanonicalPath();
} catch (IOException e) {
    UFStatus.handle(Priority.PROBLEM, Activator.PLUGIN_ID, "Colormaps",
        null, "Error determining colormap dir", e);
}
```

- This snippet shows a typical usage of *UFStatus*
 - When any error occurs, *UFStatus.handle()* is called in the error handler
- *UFStatus.handle* arguments are
 - Message priority – one of the *UFStatus.Priority* values
 - Name of the component sending the message, normally *Activator.PLUGIN_ID*
 - Category of the message
 - Subcategory of the message – may be *null* if no subcategory
 - The text of the message
 - (optional) The *Throwable* that generated the message



Questions?



Wrap-Up



Summary

- Covered CAVE platform updates
- Covered additional capabilities added to CAVE
- Covered CAVE baseline reorganization
- Covered CAVE serialization improvements
- Covered modified error-handling capability



Resources

- On the ADE TO10 DVD
 - Current code available for examination in the ADE baseline
 - JavaDoc documentation available
- Also available
 - TO10 Training Updates
 - TO-T1 Training Materials

