Raytheon

Customer Success Is Our Mission

AWIPS Continuous Technology Refresh (CTR)

AWIPS Software CTR

TO10: Outbrief

February 12, 2008

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 this restriction are contained in all sheets.

T010 Outbrief Introduction

- AWIPS T010 Summary
- AWIPS Architecture Update
- AWIPS Other Migration Capabilities
- AWIPS Hydro Status
- AWIPS GFE continued development
- AWIPS Stability Testing
- AWIPS Continuing Architecture Evolution

TO10 AWIPS-II Task Order

Raytheon

Summary

Areas addressed by the task order

- Common (WFO & RFC) hydro capability
- Hydro data ingest and Metar2Shef into ESB architecture
  - HydroView, HydroBase, TimeSeries … into CAVE
  - Added launching SSHP, RiverMonitor, and Logistical Measures
  - MPE editor into CAVE and wrapped MPEfieldgen
  - RiverPro Dialog into CAVE and wrapped Rpfengine
- Skew-T to include NSHARP features
- Command line interfaces
- Reengineered Guardian plus added a couple of features
- Reengineered TextDB with dedicated service
  - Added command line interface for TextDB, uEngine, and subscription (trigger)
  - Added 30 TAF
- Fixed numerous DRs with GFE, SmartTools, Procedures, and Formatters
- Made many updates to WarnGen, GFE hazards, and AvnFPS
- Prototype ORPG interface into new RadarServer
  - Added ingest and rendering for Radar Graphics

TO10 Major Architectural Improvements

Changes made to improve performance and maintainability

Raytheon

Summary

- Restructured the EDEX software to a OSGI organization
- Simplified the EDEX plugin pattern by using Java "Annotations" instead of XML for metadata definitions
- Extending purging flexibility by adding purging to the plugin pattern
- Simplified software builds by replacing JIBX XML serialization with JaxB
- Improved the CAVE to EDEX data interface performance with dynamic data serialization pattern (Hydroview driven)
- Improved stability by replacing the MULE ESB with CAMEL
- Simplified deployment by having JMS/CAMEL/Jetty(web) in one JVM
- Improved clustering performance with new HDF5 locking

AWIPS migration using OSGi standards

Improved standardization of Open Services Gateway Initiative OSGi

Raytheon

Summary

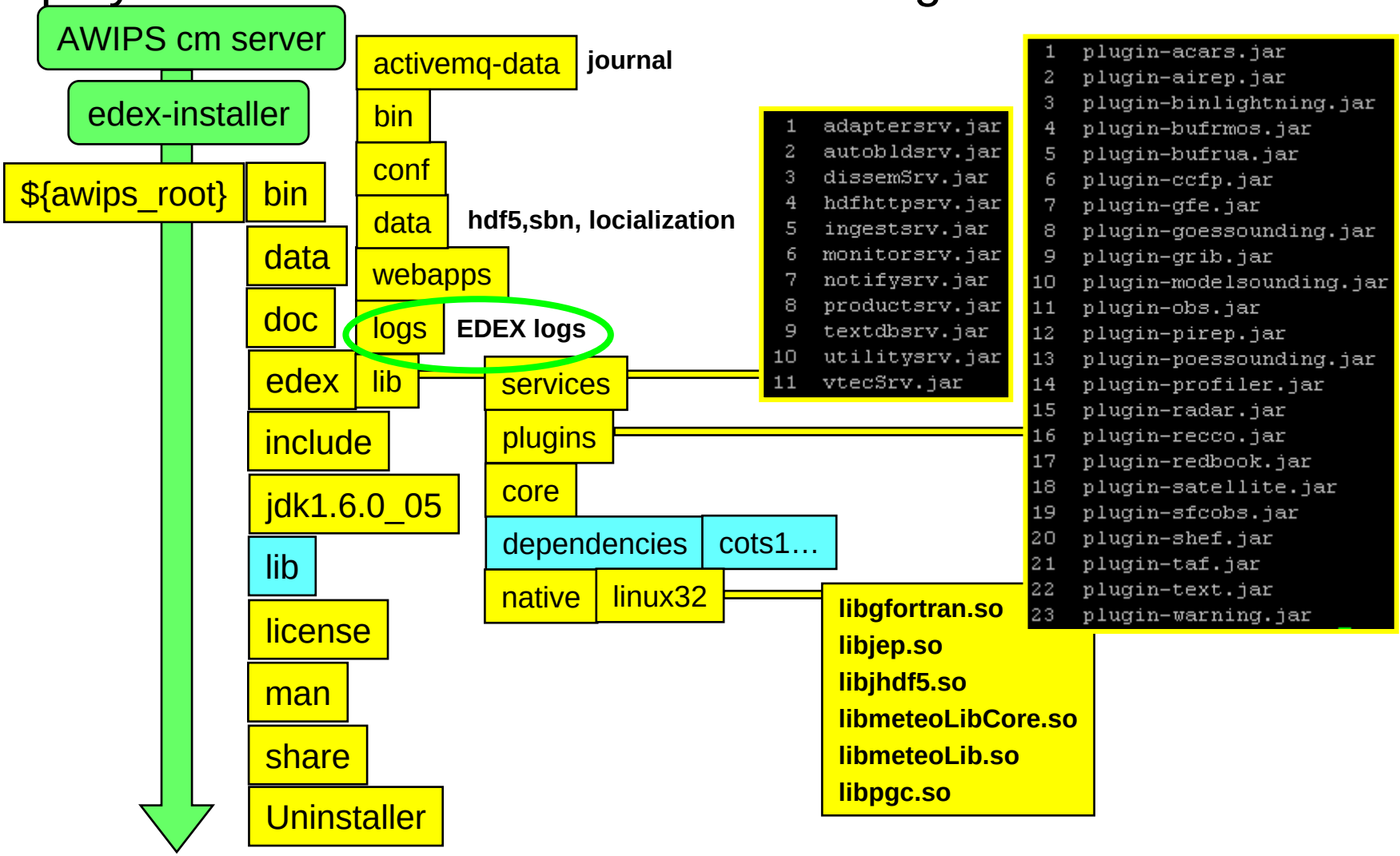
- Leveraging the OSGi deployment infrastructure
- Entire EDEX software baseline refactored to meet OSGi standards
- Each component in EDEX is an Eclipse compatible plugin project
  - Enables developers to have automatic compile when developing in the Eclipse IDE
  - Increases developer efficiency
- EDEX components will be ready for auto configuration management when the ESB container is ready for it

AWIPS Simplified Server Deployment
Everything required is controlled by CM repository

Raytheon

Summary

- Deployed libraries use tree structure to organize cots etc.

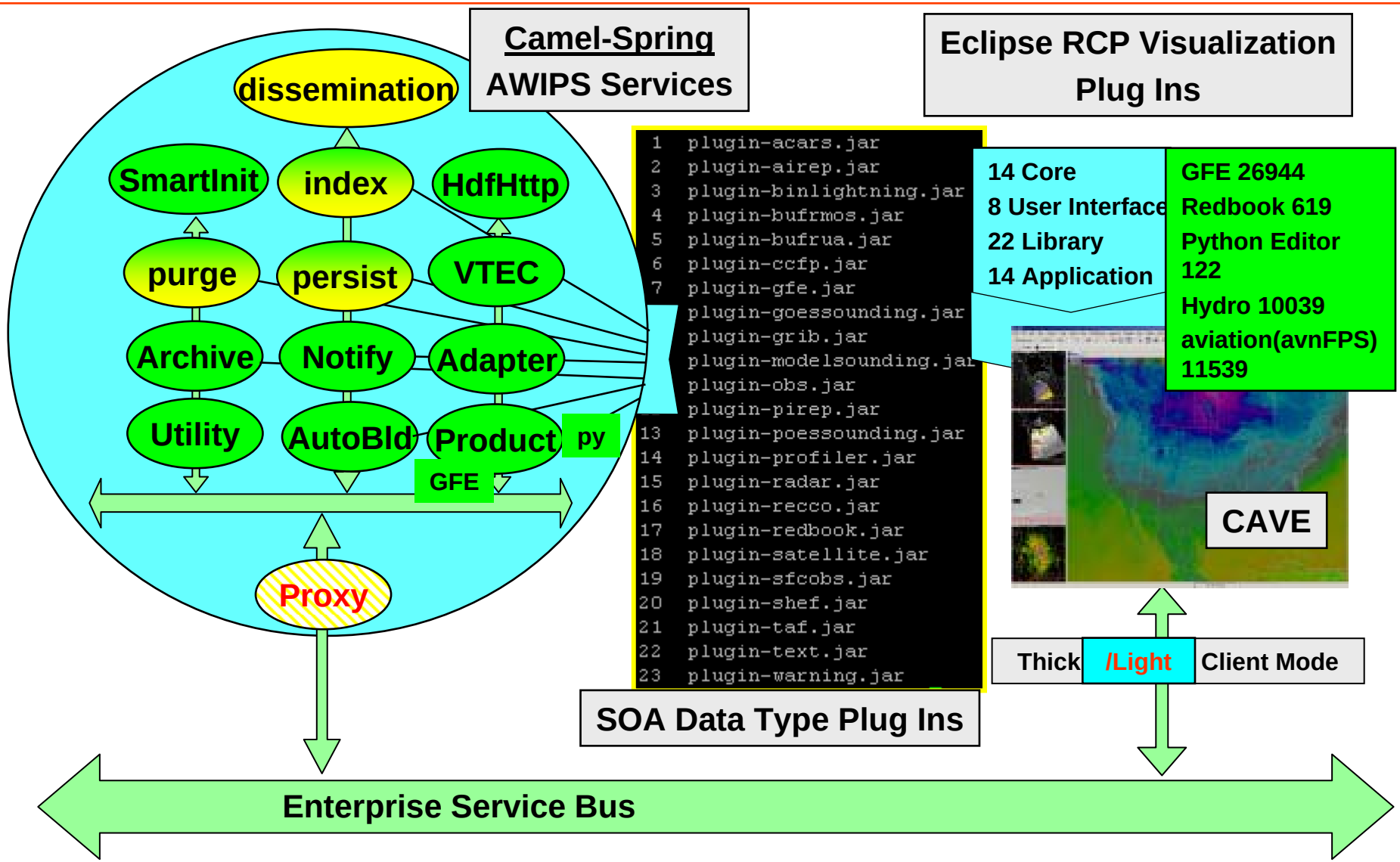


AWIPS Services and Plug Ins

ESB SOA Plug In Adaptable

Raytheon

Future Architecture



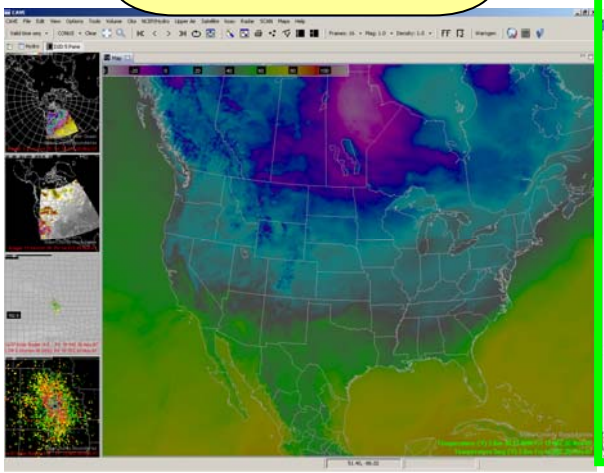
TO10 CAVE Eclipse Perspectives
AWIPS has 5 perspectives with unique GUIs

Raytheon

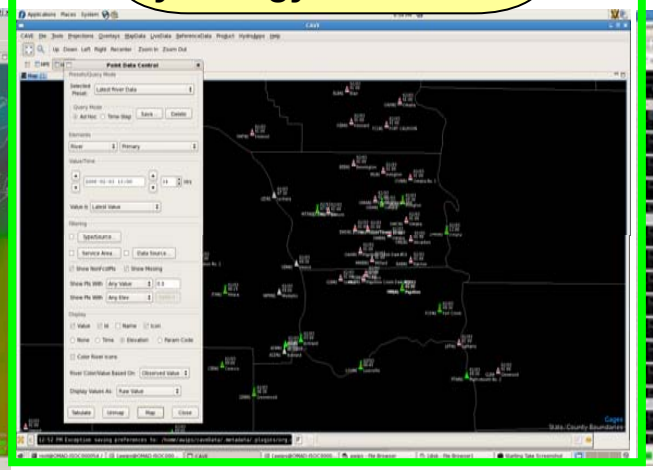
Architecture

- Release 1 implement box behavior during perspective switching

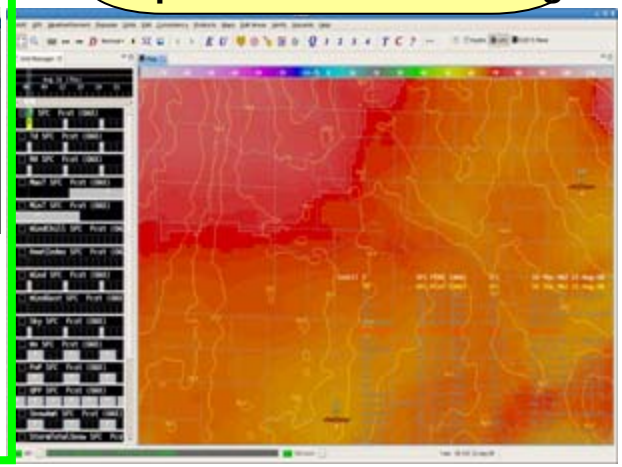
Weather Workstation



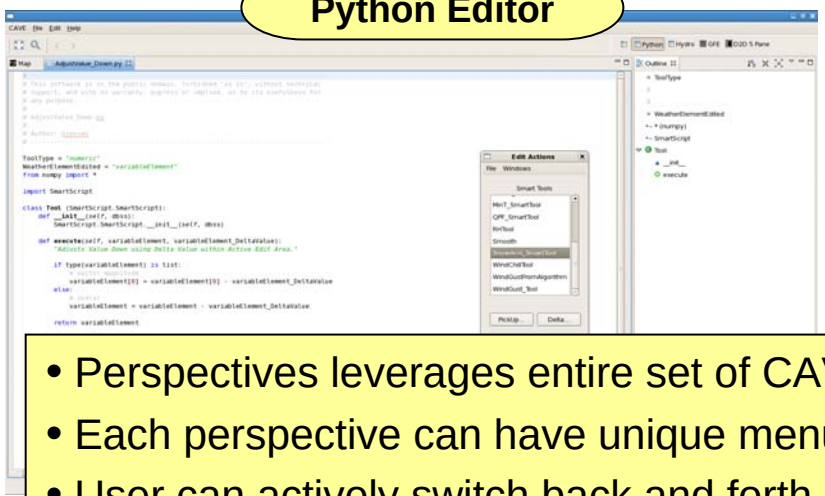
Hydrology Workstation



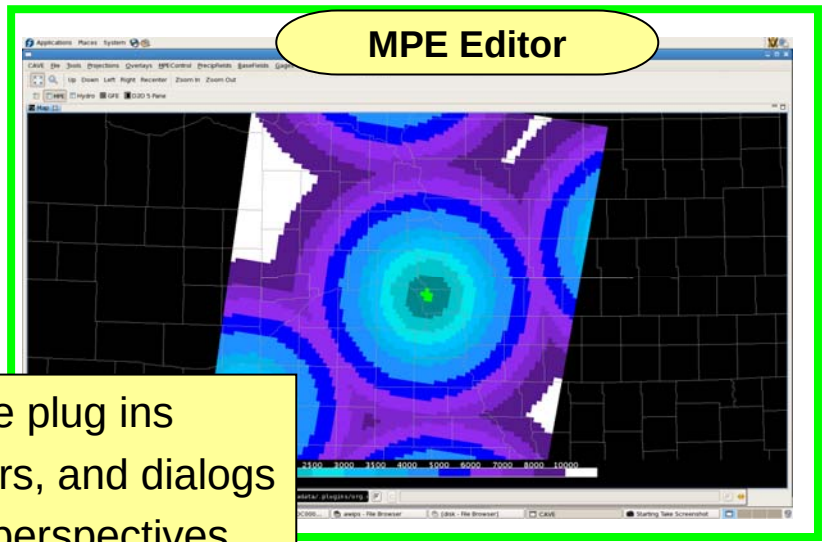
Graphical Forecast Editing



Python Editor



MPE Editor



- Perspectives leverages entire set of CAVE eclipse plug ins
- Each perspective can have unique menus, toolbars, and dialogs
- User can actively switch back and forth between perspectives

AWIPS TO10 Camel Results Summary
Conversion is complete with many benefits to AWIPS

Raytheon

Architecture

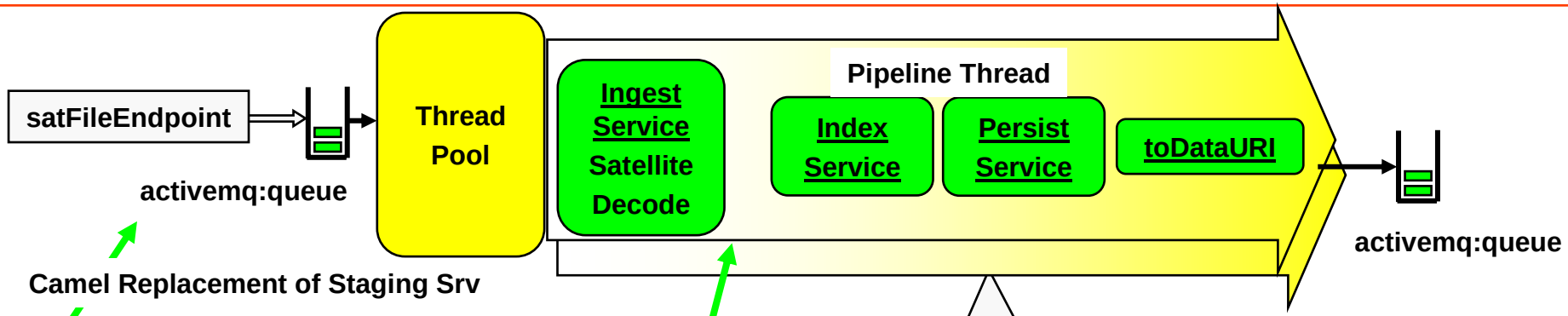
- We evaluated Apache Camel evaluated under heavy data loads from all the plug-ins
- Handles large spikes in data flow without memory overruns
  - Better messaging efficiency
    - activeMQ and the ESB run in the same JVM
    - JMS automatically switches to virtual memory queues for internal messages
    - Greater control of ingest processing through execution pipelines
    - Better management of threads through thread pools with less thread context switching
    - Simplified startup (only one JVM process instead of two)
  - Uses the SPRING dependency injection container
  - AWIPS services become simple POJOs without any ESB dependencies
  - Simplified XML ESB wiring
    - Wiring XML packaged within the plugin JAR file
  - http://activemq.apache.org/camel/index.html
- Log4j will continue to be used but the location of log files has changed
  - ./edex/logs

Camel ESB Wiring Packaged in Plug In

Simplies plug in deployment

Raytheon

Architecture



Camel Replacement of Staging Srv

Example: Satellite Ingest

```
<!-- Begin Sat routes -->
<route id="satFileConsumerRoute">
    <from ref="satFileEndpoint"/>
    <bean ref="fileToString" method="absPath"/>
    <setHeader headerName="pluginName">
        <constant>satellite</constant>
    </setHeader>
    <to uri="activemq:queue:Ingest.Generic"/>
</route>
<route id="satIngestRoute">
    <from uri="directvm:satelliteIngest"/>
    <multicast parallelProcessing="false">
       <try>
          <pipeline>
             <bean ref="stringToFile" method="toFile"/>
             <bean ref="satDecoder" method="decode"/>
             <to uri="directvm:persistIndexAlert"/>
          </pipeline>
          <catch>
             <exception>java.lang.Throwable</exception>
             <to uri="log:sat?level=ERROR&amp;showBody=fal
          </catch>
       </try>
    <bean ref="deleteFile" method="delete"/>
    </multicast>
</route>
```

- Pipeline out of thread pool <u>decreases thread switching</u> increasing performance
- Thread Pool <u>enable fine grain tuning</u> and enables high priority data to pass through

Improve HDF5 locking under high transaction rates

- Changing the HDF5 creation to each cluster member having it's own file
  - Reduces file contention
  - Metadata keeps track of files
- Added new HDF5 locking concept because of problems with NFS and async
  - Can set NFS configuration the same as AWIPS-I
  - Using flock for Java NIO which forces a lock through NFS
- Can configure NFS to use noasync option for improved NFS performance

TO10 ESB Serialization Improvements

Core performance problem in current ESB technology

Raytheon

Architecture

- Replaced the existing JiBX serialization technology
  - Binary Serialization using Thrift
  - XML Serialization using JAXB 2.1
- Implemented purely at runtime using state-of-the-art technologies to maintain fast performance
  - No post-compile steps
  - Descriptions of serialization implemented purely in Annotations
- Uses binary in communication between CAVE and EDEX, XML in user-interface situations
- Why JAXB 2.1? JiBX could not handle the return sets for HydroView and Plots
  - Included in JDK 6
  - Has EXTENSIVE built in class support for HashMaps, Lists, etc. (better support than JiBX)
  - Like JiBX, works with POJOs, not class generation
  - Other considered projects: XStream, XMLBeans, others.

TO10 Serialization Improvements

- Why Thrift?
  - Thrift is open standard. Developed by Facebook, currently an Apache Incubator project.
  - Simple and easy to understand.
  - API available in variety of languages
  - Was easily modified to become “self-describing”
  - Other considered projects: Google Protocol Buffers (had compilation requirements), NetCDF4/HDF5 (not suited for wire protocols), Fast Infoset (Performance concerns)

AWIPS TO10 Service Message Serialization

SOA Dynamic Binary Serialization Increases Performance

Raytheon

Architecture

Simplifies Coding

Java Class with Serialization Annonations (i.e. SatelliteRecord.java)

End Point URL for Thrift | **ThriftTransformer**

End Point URL for XML | **JaxbTransformer**

DynamicSerializationManager
- Runs at runtime
- Uses anonations
- Caches output

On the fly code generation makes Reflection operations fast

CGLib (hibernate) code generation

Binary Response Payload

XML Response Payload

<<CAMEL ESB Wiring>>

Exposed to Developer through ESB transformer Pattern

```
<camelContext id="uengine-camel" xmlns="http://activemq.apache.org/camel/schema/
    <!-- UEng
    <route id
        <from uri="jettyedex:http://0.0.0.0:9581/services/pyproductthrift" />
        <bean ref="uEngine" method="executePython" />
        <bean ref="serializationUtil" method="transformToThrift" />
    </route>
            <route id="uEngineHttpJaxb">
            <from uri="jettyedex:http://0.0.0.0:9581/services/pyproductjaxb" />
        <bean ref="uEngine" method="executePython" />
        <bean ref="serializationUtil" method="marshalToXml" />
            </route>
            <route id="uEngineJmsThrift">
        <from uri="activemq:queue:uEngineThrift" />
        <bean ref="uEngine" method="executePython" />
        <bean ref="serializationUtil" method="transformToThrift" />
    </route>
            <route id="uEngineJmsJaxb">
        <from uri="activemq:queue:uEngineJaxb" />
        <bean ref="uEngine" method="executePython" />
        <bean ref="serializationUtil" method="marshalToXml" />
    </route>
```

- **Thrift is normally static and runs at compile to produce java classes**
- **T010 added dynamic serialization based on annotations for greater control of data that is transferred**
- **T010 added caching of generated objects so that the generation of classes at a one time hit**

TO10 Serialization Improvements
CAVE to EDEX improved interface performance

Raytheon

Architecture

- Spot check of performance benchmarks
  - Test #1: Serialize and Deserialize 1000 “typical” MetarRecords
    - **JiBX**: 752ms Serialize / 1090ms Deserialize
    - **Dynamic Thrift:** 550ms Serialize / 801ms Deserialize
  - Test #2: Serialize and Deserialize Object with float[] of size of grid218.
    - **JiBX:** 240ms Serialize / 310ms Deserialize
    - **Dynamic Thrift:** 42ms Serialize / 38ms Deserialize

SOA datatype Plug In Enhancements
Field extendable decoder model using Python

Raytheon

Architecture

i.e. Edex Warning Plug In

- Enables reuse of GFE "VTECdecoder.py"
- Enables decoder changes and extensions to deployed systems

EDEX Plug In Core Interfaces

Java Interface To PythonDecoder

PythonNumpyByteArray

WarningDecoder.py

WarningRecord

```
@Entity
@Table(name="warning")
@XmlRootElement
@XmlAccessorType(XmlAccessType.NONE)
@DynamicSerialize
public class WarningRecord extends PluginDataObject {

    private static final long serialVersionUID = 1L;

    @Column(length=32)
    @XmlAttribute
    @DynamicSerializeElement
    private String wmoid;
```

<Metadata Table> Warning

Note: Serialization Annonations

AWIPS TO10 Plug In Enhancements

Simplified metadata definitions through Java annotations

Raytheon

Architecture

Java Plugin Source Code + annotations → **MetaData Table Creation**

i.e. Example

SatelliteRecord.java

```
@Entity
@Table(name = "satellite")
@XmlRootElement
@XmlAccessorType(XmlAccessType.NONE)
@DynamicSerialize
public class SatelliteRecord extends PluginDataObject
        ISpatialEnabled {

    private static final long serialVersionUID = 1L;

    /**
     * The source of the data - NESDIS
     */
    @Column(length=31)
    @DataURI(position = 1)
    @XmlAttribute
    @DynamicSerializeElement
    private String source;
```

satellite Table in Metadata DB

datauri [PK] character varying(255)	fo in	fc bc	re ti	ut ch	ra ti	ra ti	in ti	cr ch	nu in	pl ch	sa in	sa re	sat rea	se ch	siz int	source character var	ur ch	up re	up re	cc in
/satellite/2008-11-25_09:22:15.0/NESDIS/Mi	0	FA	20	[]	20	20	20	Mi	43	Re	0	0	0	Al	504	NESDIS		0	0	-1

Eliminated PostgresSQL Table Partitions to improve query performance

Hibernate Annotations
Note: Same style as JavaDoc

- **Using Hibernate annotations eliminates two XML files from each plugin**
- **Simplifies the creation / maintance of plugin metadata**

AWIPS T010 Improved Purging Flexibility

More flexible metadata purging to handle versioning

Raytheon

Architecture

Added new interface to plug-in pattern to enable specific plug-ins to have custom purging rules

Default plug-in purging based on retention time stored in the plug in registry table

Quart Timer Once / hour

PurgeSrv

<java>
DefaultPurgerImpl

<java>
GfePurger

GFE Plug In

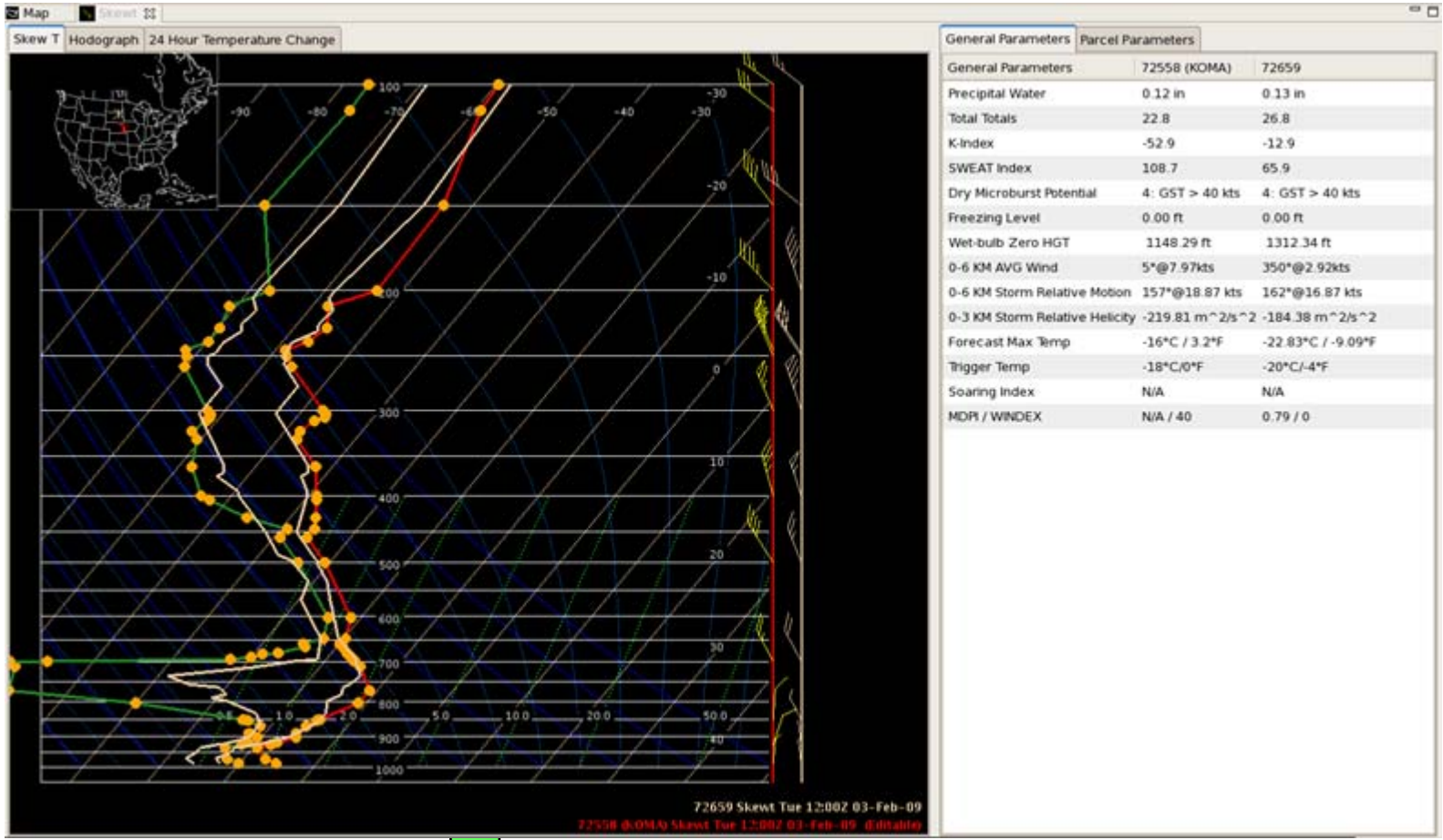
<plugin registry>
Plug_Info Table

name [PK] characte	database character var	initialized boolean	retentiontime integer	tablename character var	version real
airep	metadata	TRUE	24	airep	1
binlightning	metadata	TRUE	24	binlightning	1
bufrmos	metadata	TRUE	24	bufrmos	1
bufrua	metadata	TRUE	24	bufrua	1
ccfp	metadata	TRUE	24	ccfp	1
gfe	metadata	TRUE	24	gfe	1
goessounding	metadata	TRUE	24	goessounding	1
grib	metadata	TRUE	24	grib	1
modelsounding	metadata	TRUE	24	modelsounding	1
obs	metadata	TRUE	24	obs	1
pirep	metadata	TRUE	24	pirep	1
poessounding	metadata	TRUE	24	poessounding	1
profiler	metadata	TRUE	24	profiler	1
radar	metadata	TRUE	24	radar	1
recco	metadata	TRUE	24	recco	1
redbook	metadata	TRUE	24	redbook	1
satellite	metadata	TRUE	24	satellite	1
sfcobs	metadata	TRUE	24	sfcobs	1
shef	ihfs	TRUE	24	<unspecified>	1
taf	metadata	TRUE	24	taf	1
text	fxa	TRUE	24	<unspecified>	1
warning	metadata	TRUE	24	warning	1

T010 AWIPS Skew-T
Combined features of NSHARP and D2D

Raytheon

Migration



General Parameters	72558 (KOMA)	72659
Precipital Water	0.12 in	0.13 in
Total Totals	22.8	26.8
K-Index	-52.9	-12.9
SWEAT Index	108.7	65.9
Dry Microburst Potential	4: GST > 40 kts	4: GST > 40 kts
Freezing Level	0.00 ft	0.00 ft
Wet-bulb Zero HGT	1148.29 ft	1312.34 ft
0-6 KM AVG Wind	5*@7.97kts	350*@2.92kts
0-6 KM Storm Relative Motion	157*@18.87 kts	162*@16.87 kts
0-3 KM Storm Relative Helicity	-219.81 m^2/s^2	-184.38 m^2/s^2
Forecast Max Temp	-16*C / 3.2*F	-22.83*C / -9.09*F
Trigger Temp	-18*C/0*F	-20*C/-4*F
Soaring Index	N/A	N/A
MDPI / WINDEX	N/A / 40	0.79 / 0

CAVE Plugin

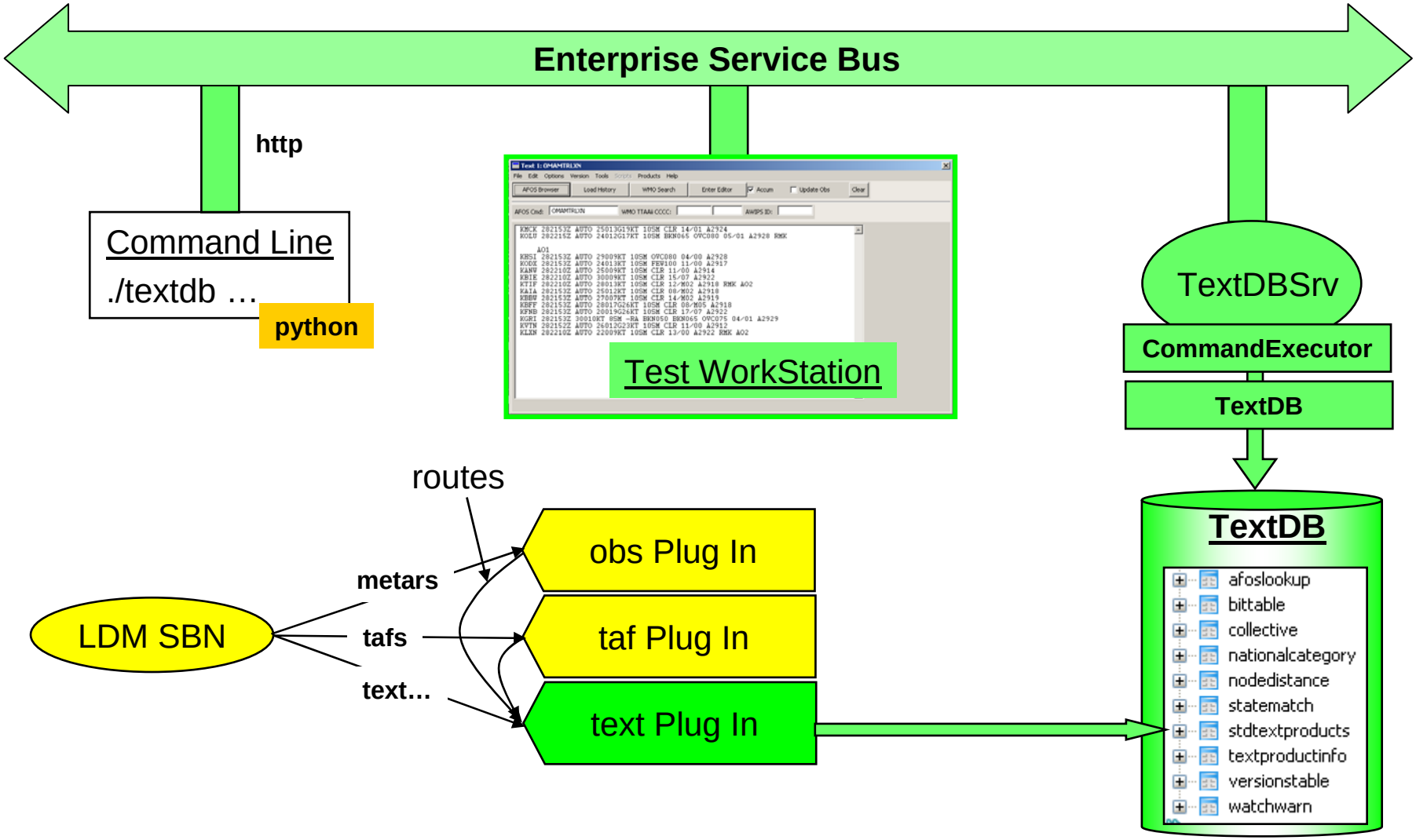
.viz.skewt.ui

TO10 TextDB

New service, DataType Plugin, TextWS interface

Raytheon

Migration

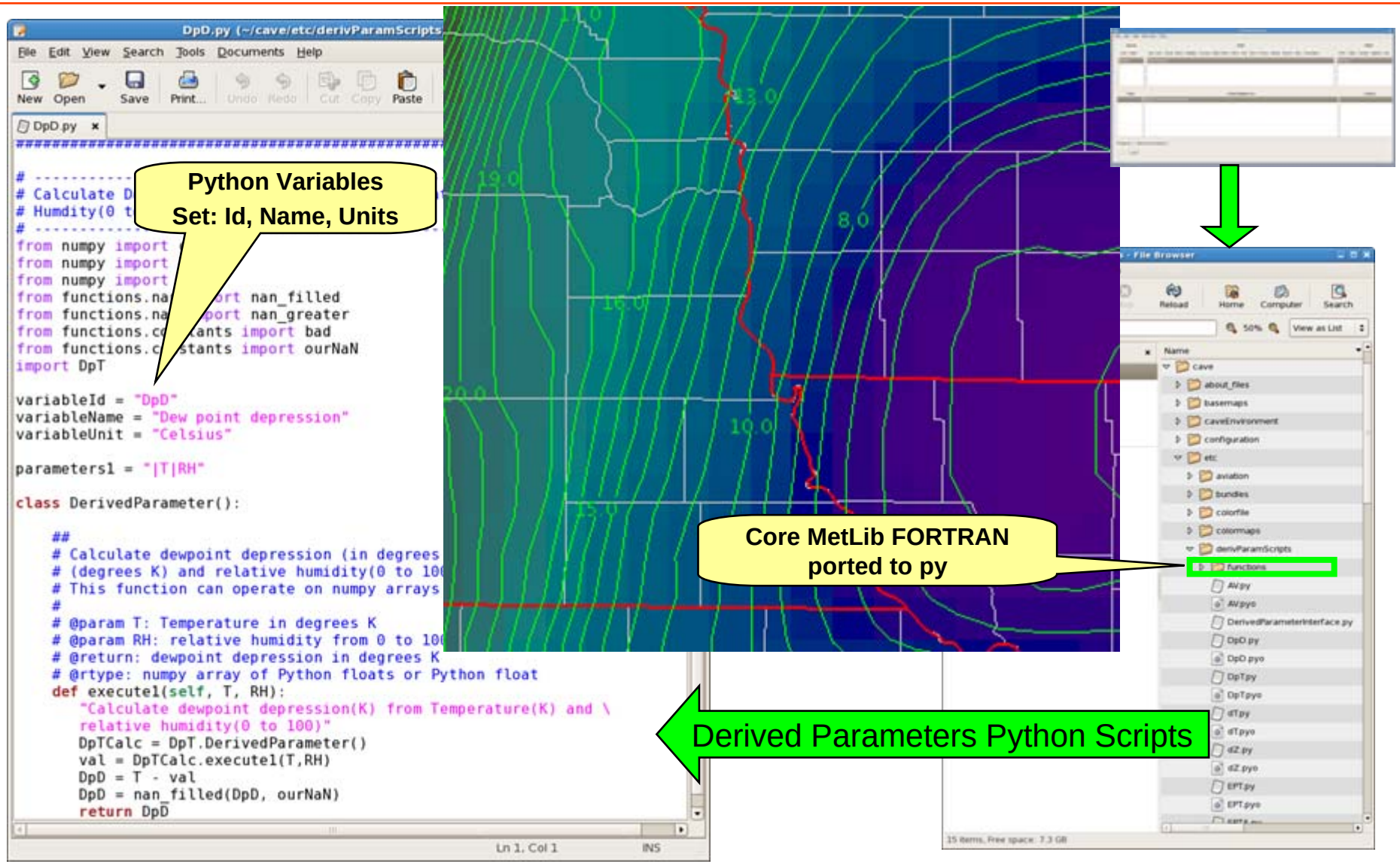


Extend TO9 Derived Parameter Pattern

Larger range of data types

Raytheon

migration



TO10 DataCubeContainer Enhancements

migration

New Implementation:

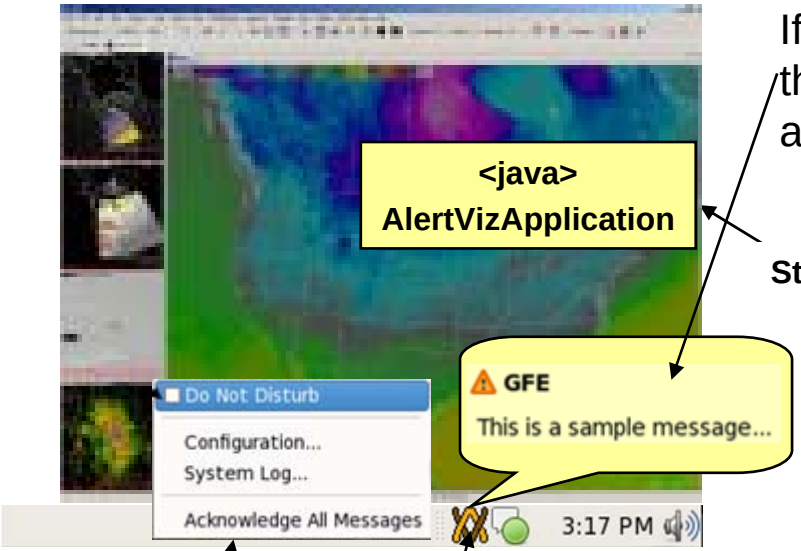
- Point and line based derived parameters implemented within the container
- GRIB specific code will be moved into the GRIB plugin
- Missing capabilities including the 3D cube, the Slice method, and level definition support will be implemented to complete derived parameter capability
- Additionally, the static calculations for dx, dy, and Coriolis need to be completed and verified
- More work in TO11 to facilate Levels efficiently (proposal details out the list of parameters and levels)
- New for T011…

Gaurdian for Status and Logs

Centralizes status monitoring for all applications

Raytheon

migration



If the Layout (category) is set to system tray in the configuration dialog, incoming message will appear in a message balloon.

<java>
AlertVizApplication

Stomp Message Protocal

<Python>
sendNotificationMsg
(command line utility)

Command line application allows you to send a message with a specified Category, Source, Priority, and message text.

Alert Visualization icon in the system tray.

Right-clicking the system tray icon will being up a popup menu.

- **Do Not Disturb – Prevents the Alert Popup dialog from appearing. Any messages to be acknowledged will cause the system tray icon to flash.**
- **Configuration… - Shows the configuration dialog.**
- **System Log… - shows the System Log with all of the recent messages.**
- **Acknowledge All Messages – Acknowledge all of the Alert messages without bringing up the Alert Popup dialog.**

TO10 Command Line Interface
Supports local application migration and development

Raytheon

migration

- TO 10 introduces a new AWIPS II component, the Command Line Interface (CLI) tools
  - Three CLI tools are provided:
    - **textdb** – this is a rehost of the existing (AWIPS I) textdb tool

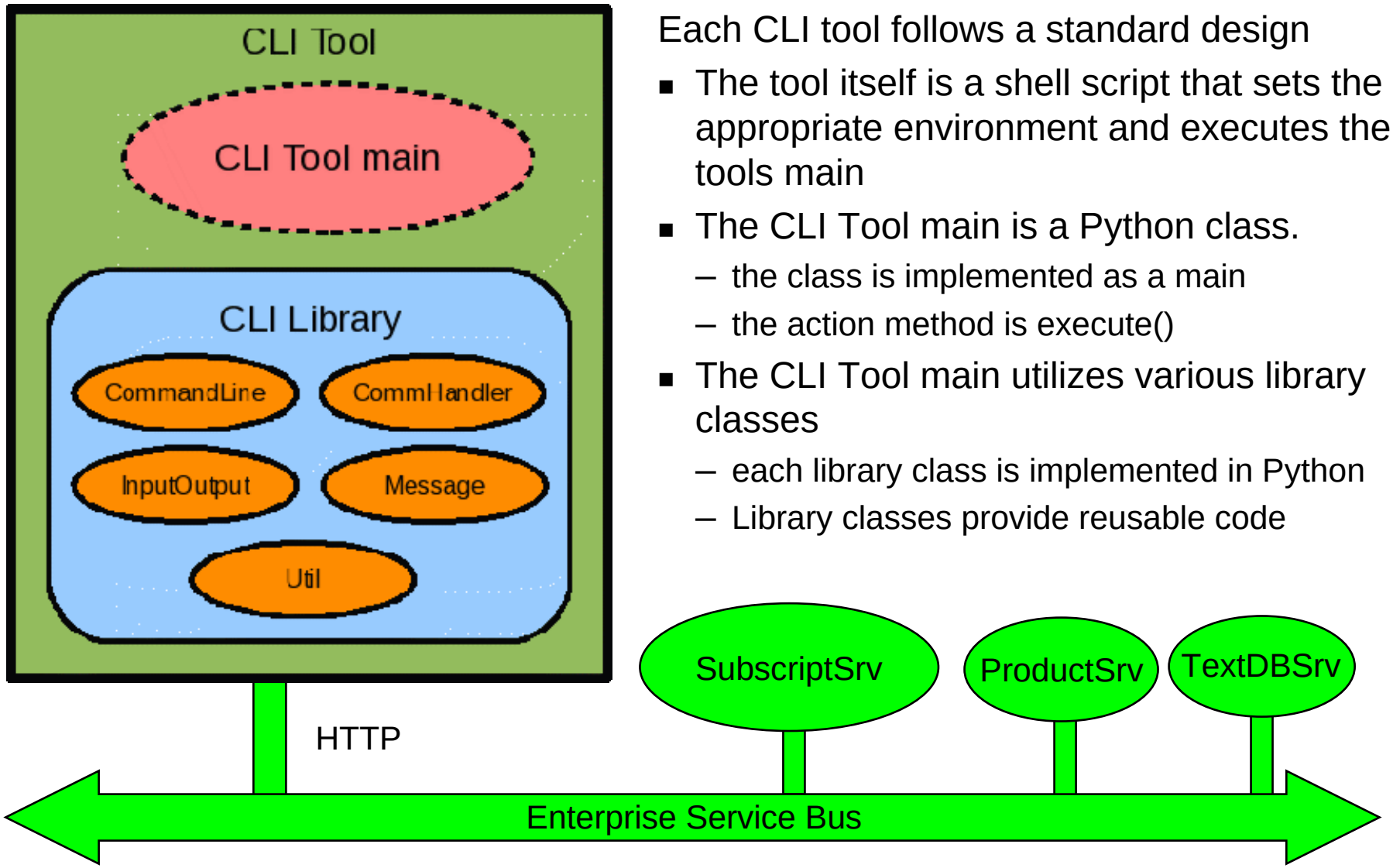
      existing textdb command line flags and responses have been preserved
    - **uengine** –a command line tool that interacts with the EDEX Product Server to run Micro Engine scripts
    - **subscription** – a command line tool that interacts with the EDEX Subscription Service to manage product sucscriptions
  - CLI tools are packaged in a separate installer provided with the ADE
    - included in the *com.raytheon.uf.tools.cli* project
    - The tools only have a dependancy on python
  - TO10 Programmers briefing has a lot on detail on these tools

TO10 Command Line Interface – Design

Provides familiar mechanisms for the local app developers

Raytheon

migration



Each CLI tool follows a standard design

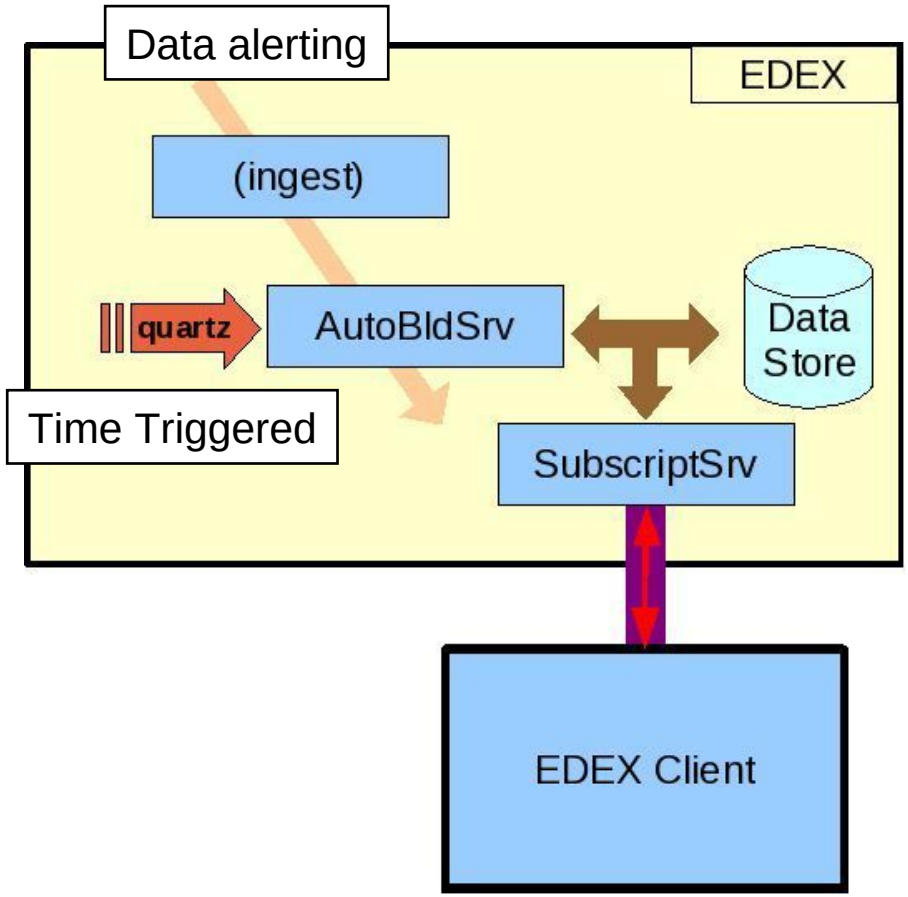
- The tool itself is a shell script that sets the appropriate environment and executes the tools main
- The CLI Tool main is a Python class.
  - the class is implemented as a main
  - the action method is execute()
- The CLI Tool main utilizes various library classes
  - each library class is implemented in Python
  - Library classes provide reusable code

T010 Subscription Service Design Concept
Notification and triggers alternative

Raytheon

migration

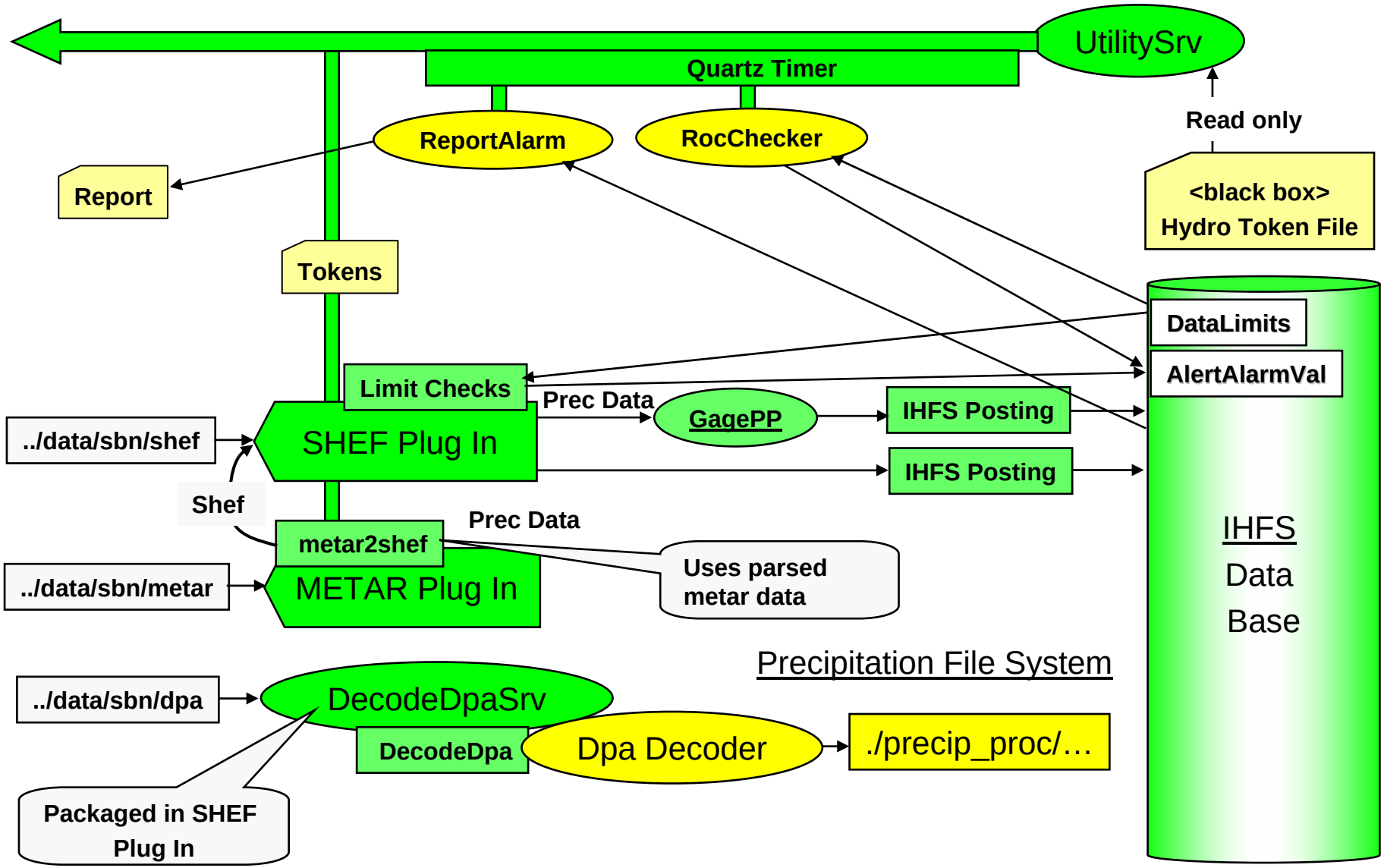
- This diagram shows design of the subscription management functionality
  - the Subscription Manager is merged into NotifySrv and renamed SubscriptSrv
  - an EDEX client interacts directly with SubscriptSrv to manage subscriptions
  - SubscriptSrv interacts with the data store and passes messages to the script runner (AutoBldSrv)
  - AutoBldSrv will be refactored to allow multiple script runners as well as product and timer based execution



T010 Hydro Ingest Processing
Gage Precip Ingest & Alert Alarm Generation

Raytheon

Hydro

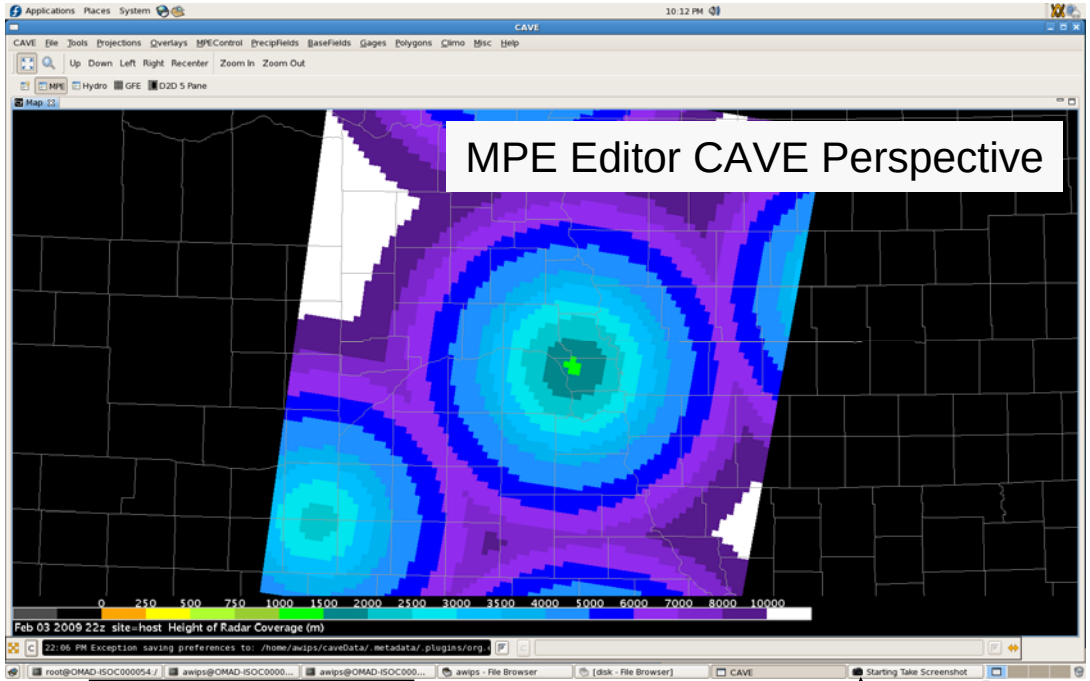


TO10 Cave MPE Perspective

MPE GUI and Business Logic + Wrapped MPEfieldgen

Raytheon

Hydro



Quart Timer Once / hour

CAVE Plug Ins

.viz.mpe

.viz.mpe.ui

http

MpeFieldGenSrv

MPEfieldgen

NAS NFS mounted file system

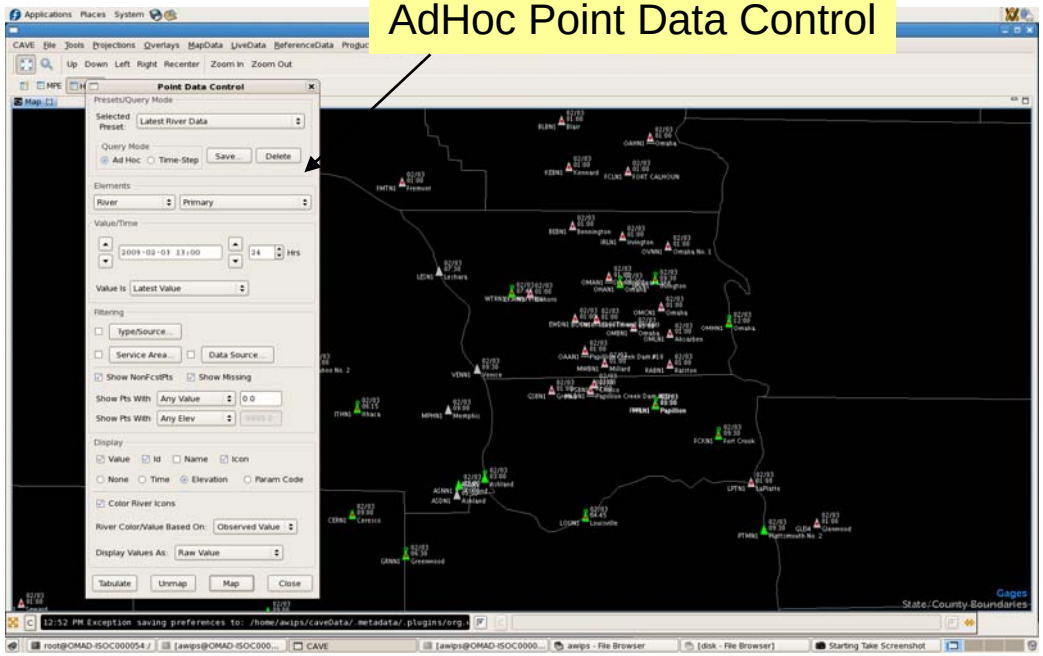
./precip_proc/…

TO10 Hydro Perspective

HydroView GUIs, HydroBase GUIs, XDAT, and others…

Raytheon

Hydro



- TO10 developed a data access approach to enable flexible access to the IHFS

CAVE Plug Ins

- .viz.hydro
- .viz.hydrobase
- .viz.hydrocommon

Enables Direct SQL/HQL

Data Managers …

DirectDbQuery

IHFS Data Base

Hydroview Dialog Status for T010 DT

Hydro

Dialog Name	Status	Dialog Name	Status
Point Data Control	Adhoc done, time step* T011	**Point Prec Accumulation**	Limited
Flash Flood Guidance	T011	**River Summary**	Done
Time Series	Done, DRs	**Station Profile**	Done
Alert and Alarm	Done	**Best Estimate QPE**	Dialog Done T011 will connect
Product Viewer	Done	**Station List**	Done
Impact Statement	Done	**Dam Display Control**	T011
Questionable and Bad Data	Done	**Data Trash Can**	Done
Data Sources	Done	**Staff Gage**	Done
Station Reporting Status	Done	**Color Scale**	Need to change color
Station Legend	Done	**About**	Done

Xnav	T011 Merging with HydroView Capabilities		
Change Colors	T011 Merging with HydroView Capabilities		
Change Precipitation Data Filter	T011 Merging with HydroView Capabilities		XNAV GUIs
Change Precipitation Thresholds	T011 Merging with HydroView Capabilities		
Flash Flood Guidance	T011 Merging with HydroView Capabilities		
Miscellaneous Data	T011 Merging with HydroView Capabilities		
RFC Gridded QPF (NMAP)	T011 Merging with HydroView Capabilities		
WFO FMAP	T011 Merging with HydroView Capabilities		

Hydrobase Dialog Status for T010 DT

Hydro

Hydrobase (Main dialog)	Done	**RiverPro Gen Parms**	Done
River Gage	Done	**RiverPro Forecast Groups**	Done
Flood Category	Done	**County/Zone UGC**	Done
Impact Statement	Done, Shared with Hydroview	**Unit Hydrograph**	Shared with Hydroview
Low Water Statement	Done, Shared with Hydroview	**Crest History**	Shared with Hydroview
Flood Damage	Done	**Benchmark**	Done
Rating Curve	Done, Shared with Hydroview	**Datum**	Done
Low Water	Done	**Purge Parameters**	Done
QC/Alert/Alarm Limits	Done	**Reference Fields**	In Work
Preferences	Done	**Areal Definitions**	T011
Add Location	Done	**Vector Definitions**	T011
Modify Location	In Work, DRs	**HydroGen Configuration**	Done
Data Sources	Done	**Contacts**	Done
Ingest Filter	Done	**Gage History**	Done
Adjustment Factors	Done	**Description**	Done
Flood Report	Data displayable via another view	**Publication**	Done
Text Report (Adam)	On Hold, T011	**References**	Done
Administration	Done	**Reservoir**	Done

Hydrobase Dialog Status for T010 DT cont.

Hydro

Dialog Name	Status		
States/Counties/Zones	Done		
Start/Stop SHEF Decoder	No, doesn't apply		
Cities	Done		
Radar Locations	Done		
NWR Transmitter Towers	Done		
TimeSeries Group Configuration	Done		
About	Done		

Need way to force config reload

Dialog Name	Status		
Edit Data	Done		
Group Data	Done		
Insert Data	Done		
Site List	Done		
Xdat	Done		
Precipitation	Done		

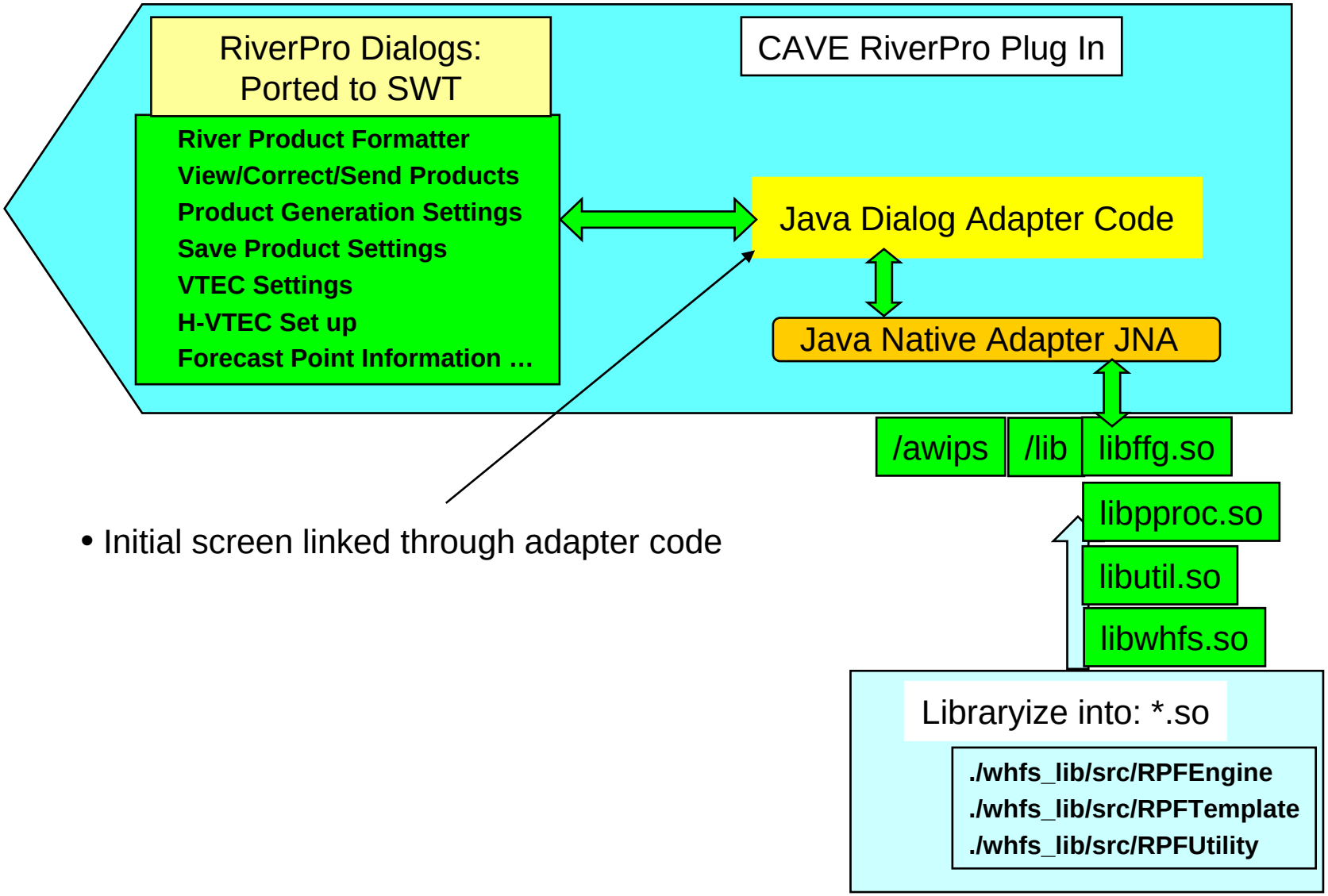
XDAT Dialog Status for T010 DT

AWIPS T010 RiverPro: Reengineering

Ported Dialogs to SWT and develop JNA interface to libs

Raytheon

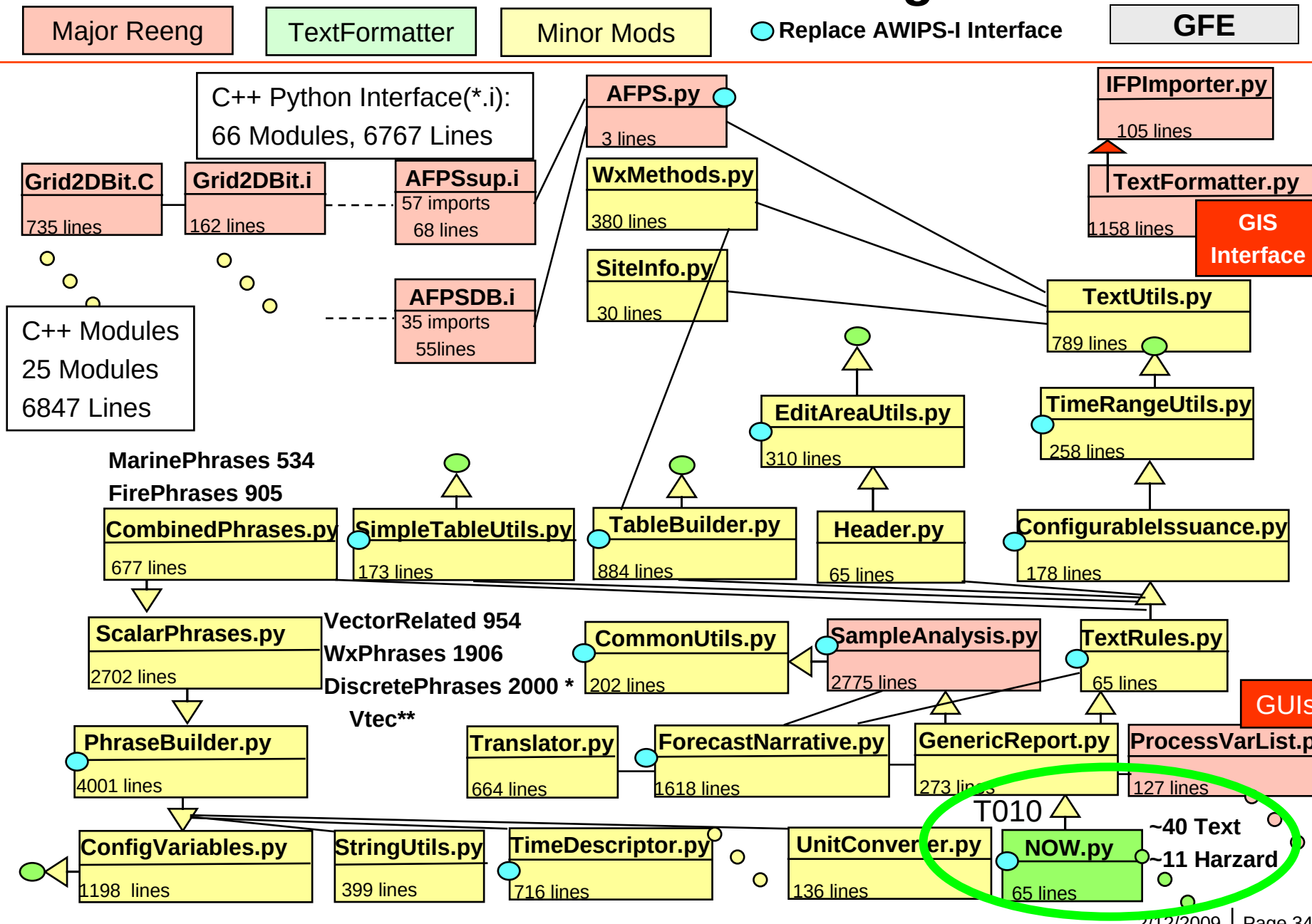
Hydro



- Initial screen linked through adapter code

TO10 TextFormatters: AWIPS-I integrated

Raytheon



Smart Tool Architecture Layers

Extended during TO10 to improve smart tool support

Raytheon

Smart Tools

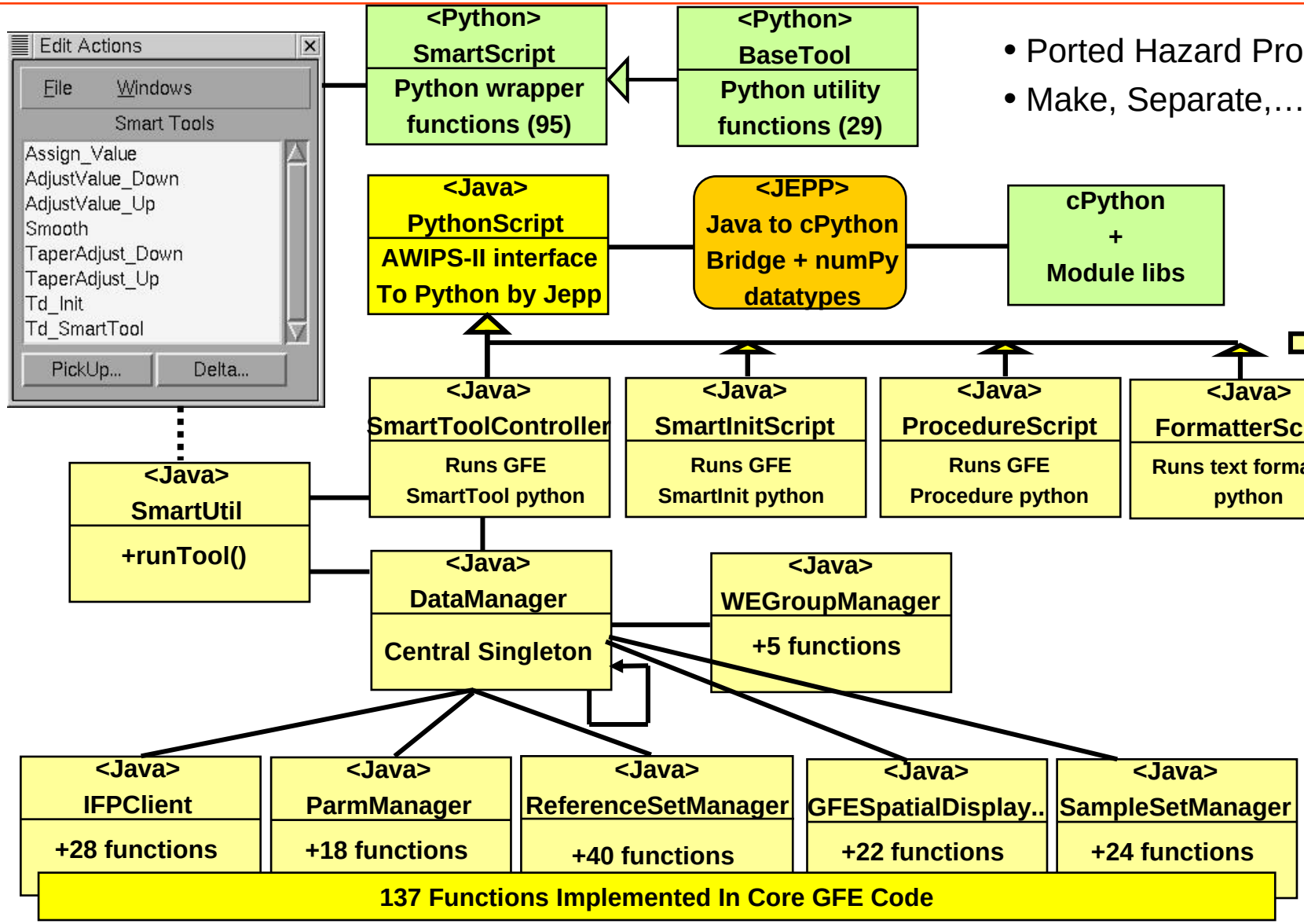
- **JEP – Java Embedded Python**
  - Modified for improved performance and numpy support
- **PythonScript – Java class that wraps JEP calls**
- **SmartToolController (new in TO10)**
  - Java class that extends PythonScript
  - Manages smart tools in a single python interpreter (enables dropped in utilities to be picked up)
- **SmartToolInterface** – python class that assists the SmartToolController with managing and running smart tools
- **Tool** – Java class ported from Tool.py that prepares and executes smart tools through the SmartToolController
- **AFPS -** is incompatible with AWIPS-II. Most the methods used in AFPS, have a equivalent Java / Python method that can do the same thing. Translating a former AFPS call is straightforward, any that are missing should be brought to our attention.
- New Java class in T010 **"GFEProcedure"** runs procedures
- TO10 - **callSmartTool** and **callProcedure** added to SmartScript.py

T010 Smart Tool Architecture Overview

Capability ready for evaluation

Raytheon

Smart Tools



- Ported Hazard Procedures
- Make, Separate,…

Converting AFPS References to AWIPS-II

Examples demonstrate ease of conversion

Raytheon

- The following is a set of examples using TimeRange and AbsTime
- Example 1 for TimeRange (Format: AWIPS I → AWIPS II)

import AFPS → import TimeRange

AFPS.TimeRange(start, end) → TimeRange.TimeRange(start, end)

- Example 2 for AbsTime (Format: AWIPS I → AWIPS II)

import AFPS → import AbsTime

AFPS.AbsTime_current() → AbsTime.current()

AFPS.AbsTimeYMD(year, month, day) → AbsTime.absTimeYMD(year, month, day)

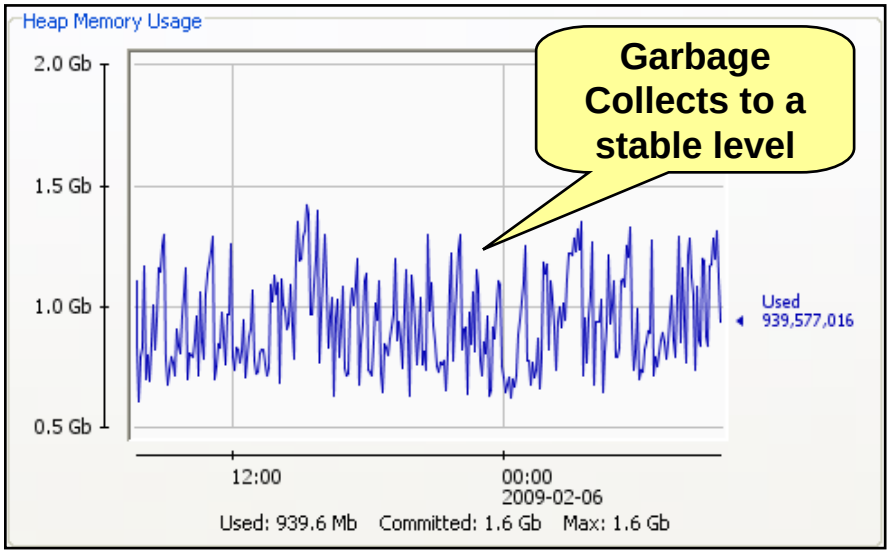
AFPS.AbsTime(time) → AbsTime.AbsTime(time)

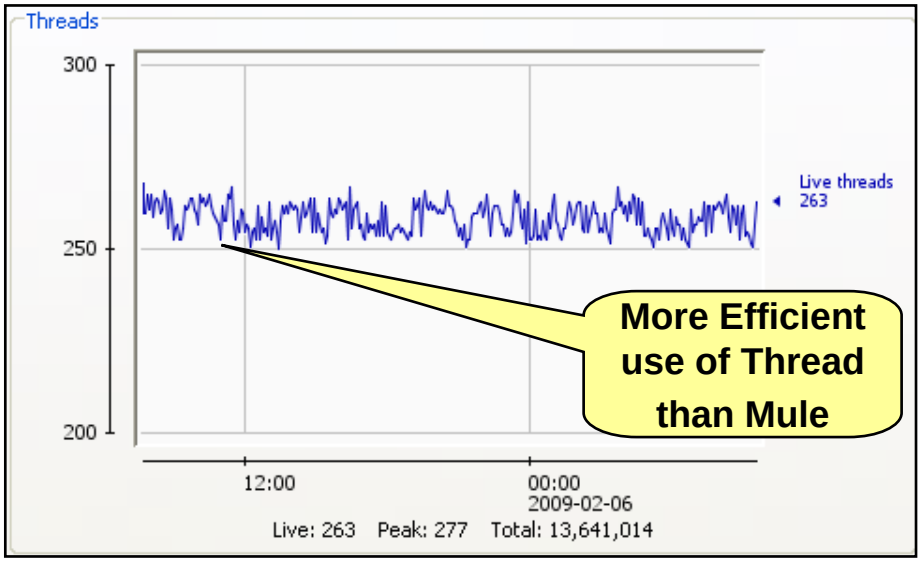
TO10 AWIPS Stability Run Results

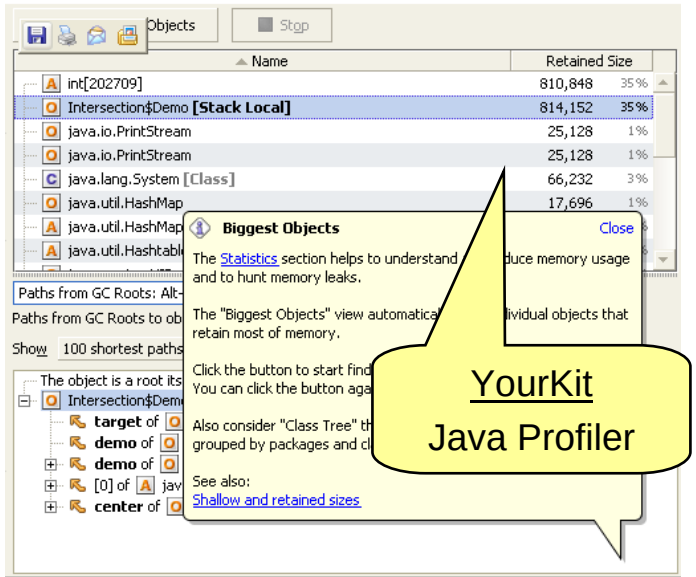
Latest build JVM 1.6.0_05

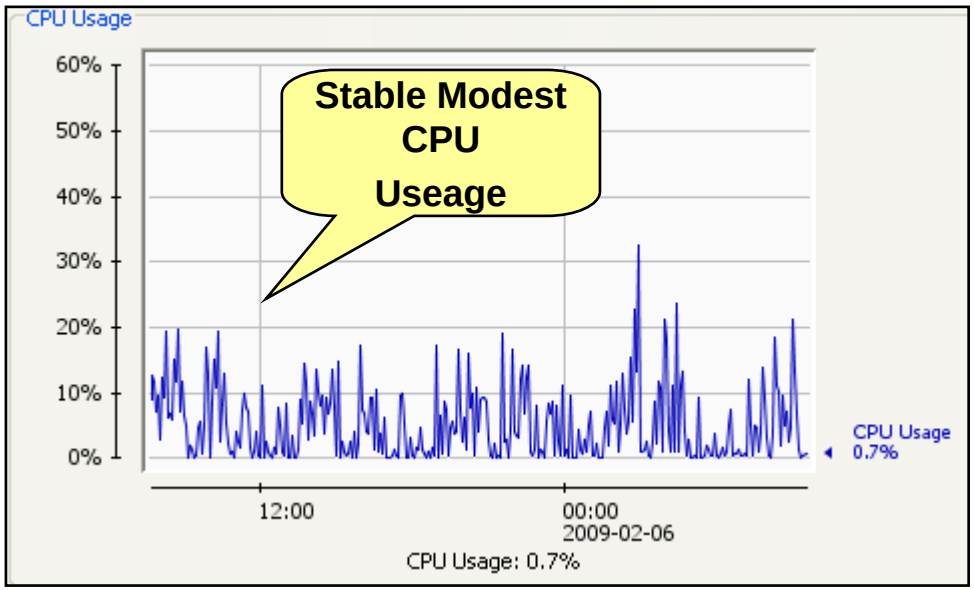
Raytheon

Stability









TO10 Delivery ReadMe
Items to evaluate with the delivery

Raytheon

ReadMe

- Hydro capabilities ready for evaluation
  - Hydro ingest: Shef, ROC, GagePP, Metar2Shef, and DPA
  - HydroView and HydroBase Capabilities except time step in HydroView
  - Timeseries display
- Core capabilities
  - Data ingest stability and performance with SBN
  - WarnGen, TestDB and TextWS
  - Display of Radar graphics and Redbook
  - Guardian replacement and its configurability
  - Command line interfaces
  - Evaluate Skew-T
- GFE related
  - Smart tools and procedures
  - Hazards generation and tools
  - Text formatters run manually
  - Retest AC001 and AC009 regression tests
- ORPG Prototype Interface
  - Software to be installed on one HQ system for use of advanced IV&V testing of the TO10 delivery

TO11 AWIPS-II Infrastructure Improvements in Progress

Raytheon

Arch. Evolution

- Improve **cluster performance** with distributed Cache
- Add **command line** interface to CAVE for GFE applications including a mechanism for running the **automated Formatter** test procedures
  - Adding a function getData() to enable retrieving “plug in” data for SmartTools
- Improve **GRIB retrieval** performance and purging flexibility
- Migrate to a pure “C” Grib1/2 Decoder from the Unidata One
  - GRIB2 decoding using J2000 is inadequate
  - Use GLUEGEN to generate JAVA interface to “C” decoder
- Reorganize CAVE core to accommodate entire range of rendering (map, cross section, xy…)
  - Simplify the pattern for adding large menu structures to CAVE
- Improve the performance of point data retrievals through HDF5 structures
- Consolidate the use of maps across the applications to PostgisSQL
  - Create tools to add local map features to PostgisSQL for WarnGen etc.
- Migrate to RH 5 and update versions of the open source projects
- Improve the separation between CORE and application plug-ins