**INTRODUCTION**

Once your gridded data is ingesting and visible via the Product Browser in D2D you can set up access to this data from GFE by following the steps and information below. Please note that the ingest of non-baseline GriB datasets is outside the scope of this document.

**Table 1** details document conventions and variables used throughout; it also defines any globally applicable information.

**Table 1. Document Conventions**

|  |  |
| --- | --- |
| Convention | Explanation |
| Commands to be executed | Commands to be executed on a system are in Courier New fontface and are assumed to be executed on a single line. Formatting and command length might cause commands to wrap to a new line in this document. |
| CCC | Identifies the AW\_SITE\_IDENTIFIER under which EDEx is running as defined in **/awips2/edex/bin/setup.env**, i.e., the “localization” regardless of the GFE\_DOMAINNAME … that is, for multi-domain sites, the below configurations are to be placed in the AW\_SITE\_IDENTIFIER site-level localization tree and not the GFE\_DOMAINNAME. |
| Site level overrides | All files are complete overrides, that is, copy the file(s) from their base location to site and edit the base file making your additions. Do not create empty files under site as this will impact base definitions. There are DRs to create append overrides.  Also, all JVMs on all EDEx Processing Servers require to be bounced after changes are made. Testing can be done by bouncing a single EDEx Processing Server’s JVMs and running a smartInit with the **–h dx3** flag (or other EDEx Processing Server taken as the flag’s argument) to not pass through the edexCluster and ensure utilization of the new caches created by the edits. However, all EDEx Processing Servers are required to be updated for new model data ingest and automatic smartInit running. |

1. **Create a parameterInfo XML file**  
   /awips2/edex/data/utility/common\_static/base/grid/parameterInfo  
   /awips2/edex/data/utility/common\_static/site/CCC/grid/parameterInfo

This file is usually named similarly to the datasetID of the D2D dataset with which it is associated, but it is actually associated with the datasetID by the gfeParamInfo.xml file (see step 4 below).

The parameterInfo files contain information that was in the .cdl/.cdf files in AWIPSI.

* If you have a .cdl file from A1 you can use the **convertCDL2XML** utility to create your parameterInfo file, to convert a pre-existing AWIPSI CDL file into an AWIPSII XML. See **Table 2.**

**Table 2. convertCDL2XML Information**

| Step | Action(s) |
| --- | --- |
| Locate the CDL to convert | Most non-baseline (site-specific) CDL files are in **/data/fxa/customFiles** but they may also be in **/awips/fxa/data**. A simple find command similar to **find /awips/fxa –name “\*.cdl”** can be performed to locate any files. Note that /awips/fxa is a local mount and most likely any AWIPSI CDL files were stored on DX3/DX4 where a grids localization would’ve been executed. |
| Copy files to a staging directory | It will be easiest to stage the CDL files, however you can simply provide the location in AWIPSI as an argument to the conversion utility, as well as the output directory for the XML, when running the script in the next step. Typically a good location is **/localapps** to ensure retention of files. |
| Run the conversion utility | The convertCDL2XML exists anywhere that the **awips2-cli-${RELEASE}** RPM is installed and the script resides in **/awips2/fxa/bin**. The ncgen utility is required for the script to run, which is provided with the **netcdf** base RedHat RPM on most systems. If any requirement for the script to run is not present, the script will exit and report the error.  To run the script, assuming the CDL files to convert were copied to **/localapps/CDL2XML** and you have changed directories into **/localapps/CDL2XML**:  **/awips2/fxa/bin/convertCDL2XML –f /localapps/CDL2XML**  Simply running the convertCDL2XML without any arguments or with the –h flag will report back syntax and arguments. The default output directory will be **`pwd`/convertedCDL** (where `pwd` is the directory in which the shell is currently running when the script is executed), passing the **-o** flag followed by path defines to which the converted XML file will be stored, for example, -o /data/local will put the AWIPSII XML converted by the script into /data/local and not `pwd`/convertedCDL.  For example, to convert a CDL in /data/fxa/workFiles named GFS0p5degGbl.cdl to an AWIPSII parameterInfo XML and store that XML in /localapps one would run the following command:  **/awips2/fxa/bin/convertCDL2XML –f /data/fxa/workFiles/GFS0p5degGbl.cdl –o /localapps** |
| Verify and copy the XML | It is good practice to verify that the XML was created properly. In most instances the XML will be correct. But if the CDL is sufficiently unique (various levels for certain parameters, for example) then some XML entries may be handled improperly by ncgen. If you notice instances where CDL files are not handled properly, open a ticket with the NCF. Verify the XML against **Table 3, parameterInfo Format.**  Copy the XML file from the convertedCDL output directory to the proper location, **common\_static/site/grid/parameterInfo** and continue onto **Step 2** to complete the configuration process. |

* If you don't have a CDL file then just copy one of the baseline parameterInfo files as a template. These files are located in **edex\_static/base/grid/parameterInfo**.

Regardless of whether you are using the **convertCDL2XML** utility or copying a baseline XML parameterInfo file as a template, the important tags in the file that are required are found in **Table 3.**

**Table 3. parameterInfo Format**

| XML tag | Description |
| --- | --- |
| <fcst> | These tags define the forecast times for the model in seconds relative to the model reference time.  There should be one tag for every forecast time you wish to use in GFE.  **NOTE**: Upstream of OB13.3.1-21, the <fcst> tag is now mandatory in the parameterInfo for all datasets and for all forecast hours that may be required in GFE regardless of if they exist in the GriB data.  (See Step 4 below for more information.) |
| <short\_name> | This tag defines the parameter name as it will be seen in GFE (e.g. in WxElementBrowser using IFP as source)  It is associated the D2D parameter name via the gfeParamName.xml file  (See Step 2 below for more information.) |
| <units> | This tag defines the units of the ingested data and should normally match the value from postgres. This unit string must be parsable by the javax.measure.unit.  You can use the following SQL query to determine the appropriate unit string for a parameter while connected to dx1f:  **psql –U awips –d metadata –c “SELECT (abbreviation,unit) FROM parameter ORDER BY abbreviation”**  You can also use various other SQL statements to determine units for individual parameters, for example to determine units for potential vorticity abbreviations:  **psql –U awips –d metadata –c “SELECT (abbreviation,unit) FROM parameter WHERE abbreviation ilike ‘%pv%’ ORDER BY abbreviation”** |
| <valid\_range> | This pair of tags defines the minimum and maximum valid values for the parameter.  Values outside this range will be clamped to the min/max value. |
| <fillValue> | This tag defines the value in the grid that will be used as the “no data” value.  This value would normally lie outside the range of valid values. |
| <level> | This tag defines the level names of the data that will appear in GFE.  They are associated with the actual D2D level values by the gfeLevelMappingFile.xml (see step 3 below). |

1. **Add any new parameter names gfeParamName.xml**

/awips2/edex/data/utility/common\_static/base/parameter/alias

/awips2/edex/data/utility/common\_static/site/CCC/parameter/alias

Entries in this file are of the form:

<alias base="***d2dname***">***gfename***</alias>

* If you are adding a new model you can skip this step and the GFE parameter name will be the same as the name of the D2D parameter.
* If you want to use a different name to match existing smartInits/Tools/Procedures you can map your desired GFE parameter name to the D2D parameter name by adding an entry to the gfeParamName.xml file if one does not already exist.

**NOTE:** These mappings are not tied to a particular dataset so they affect all parameters in all datasets.

1. **Add any new levels to gfeLevelMappingFile.xml**

/awips2/edex/data/utility/edex\_static/base/grid

/awips2/edex/data/utility/edex\_static/site/CCC/grid

Entries in this file are of the form:

<Level key="gfelevel">

<DatabaseLevel levelName="***name***"[ levelOneValue="***xx***"[ levelTwoValue="***yy***"]]/>

</Level>

You can determine the appropriate D2D level values by running the following SQL query against the postgres database (on dx1f) after the model has been ingested.

Replace the value for datasetID below with the modelname being mapped, ETA218 used as an example:

**psql –U awips –d metadata –c “SELECT parameter\_abbreviation, masterlevel\_name, levelonevalue, leveltwovalue from grid\_info, level WHERE level\_id=level.id AND datasetid='ETA218' ORDER BY parameter\_abbreviation”**

* In the decoded D2D data, levels are represented by a level name, and 1 or 2 optional numeric values.
* In GFE levels are represented by a string name.
* The mapping of the GFE string names to D2D levels is contained in gfeLevelMappingFile.xml.
* The levelOneValue and levelTwoValue attributes are optional.
* The GFE level name string can be whatever you desire but is typically of the form:

levelName[ levelOneValue [ levelTwoValue ]]

1. **Add your new parameterInfo file to gfeParamInfo.xml**

/awips2/edex/data/utility/edex\_static/base/grid/parameterInfo

/awips2/edex/data/utility/edex\_static/site/CCC/parameterInfo

This file associates the parameterInfo XML file with the D2D datasetID.

Entries in this file are of the form:

<alias base="D2DdatasetID">***parameterInfoFileName***</alias>

* Copy the file from base to site and make changes to the site level file, the base file is located at **common\_static/base/grid/dataset/alias/gfeParamInfo.xml**
* If a <fcst> tag does not exist for a given forecast time in the gridded dataset’s parameterInfo file, missing timesteps may be noticed in the grid manager for that particular model.

If not all timesteps are displaying in the grid manager for a particular parameter (when compared to the time steps that are available in D2D for that same parameter), check the following to determine if there are missing <fcst> tags in the parameterInfo file:

On an EDEx Processing Server (e.g. DX3, DX4, DX5, DX6)

**grep 'No time range found for' /awips2/edex/logs/edex-ingest-smartInit-`date +%Y%m%d`.log | awk '{print $13}' | cut -d\_ -f5 | uniq**

Any dataset returned references the GFE model name mapped in the D2DMODELS array in the localConfig.py for the active GFE domain.

Assuming that the above grep command returned {MODELNAME} you can then determine the <fcst> hours that are missing via the below:

**grep 'No time range found for' /awips2/edex/logs/edex-ingest-smartInit-`date +%Y%m%d`.log | grep ${MODELNAME} | awk '{print $17}' | sort -n | uniq**

Then add these values inside the <fcst> </fcst> XML tag in the appropriate parameterInfo file.

1. **Add any accumulative parameters to D2DAccumulativeElements**

These would be defined in your localConfig.py file.

* This setting indicates to GFE that the associated D2D parameter is an accumulative element and will change how GFE determines the time range associated with the forecast hours.
* To add parameters for a new model the entry should have the following form:

**serverConfig.D2DAccumulativeElements["GFEmodelname"]=["parm1","parm2",...]**

* To add additional parameters for an existing model the entry should have the following form:

**serverConfig.D2DAccumulativeElements["GFEmodelname"]=["parm1","parm2",...]**

1. **Add the new model to D2DMODELS**

These would be defined in your localConfig.py file.

This setting associates the D2D datasetID with the desired GFE model name.

**serverConfig.D2DMODELS.append(('D2DdatasetID', 'GFEmodelname'))**

See **Table 4.**

**Table 4. GFE Relevant Configuration Files**

|  |
| --- |
| For reference, the GFE configuration file locations mentioned in this document are: |
| edex\_static/site/CCC/grid/parameterInfo/<FileName>.xml  edex\_static/site/CCC/grid/gfeLevelMappingFile.xml  common\_static/site/CCC/parameter/alias/gfeParamName.xml  common\_static/site/CCC/grid/dataset/alias/gfeParamInfo.xml  edex\_static/site/CCC/config/gfe/localConfig.py  All files should be copied from their BASE location to SITE location before editing. |