

A RiverPro Tabular Template Primer for AWIPS 5.0
by Mike Callahan, WFO Louisville KY
7/12/2001

RiverPro is an extremely powerful application that takes SHEF data stored in AWIPS and outputs it in forms suitable for the public. But because it is so flexible, the templates it uses can become a bit complex to set up. The template type that might give the largest problems is the tabular template. Rather than try to explain all the different options available, I will show what a sample template looks like to generate a routine product. With the release of AWIPS 5.0, the information in the templates has changed, so I have updated my earlier primer. Occasionally, I will refer to the RiverPro Reference Manual for more information.

A Sample Daily River Forecast Product

Here is the product we want to generate:

```
INZ089>090-KYZ035-040-049-091400-  
DAILY RIVER FORECASTS  
NATIONAL WEATHER SERVICE LOUISVILLE KY  
1130 AM EDT FRI OCT 08 2001
```

LOCATION	FS	OBS	OCT 09	OCT 10	OCT 11
<i>OHIO RIVER</i>					
MARKLAND LOWER	51	12.1	12.5	12.3	12.3
MCALPINE UPPER	23	12.2	12.2	12.2	12.2
MCALPINE LOWER	55	9.6	9.8	9.7	9.7
CANNELTON LOWER	42	10.9	10.9	10.9	10.9
<i>KENTUCKY RIVER</i>					
FORD	26	9.4	9.1	9.1	9.1
HIGH BRIDGE	30	8.3	8.9	8.9	8.9
FRANKFORT	31	7.2	6.8	6.8	6.8

END

Of course, this product has been superceded by the RVD, but it still makes a good example. I will show you how to make an RVD later. I call this product RVA_DAILY_FCST. Here is what the template in the HEADER.TPL looks like:

```
# HEADER SECTION TEMPLATES  
#  
name:rva_daily_fcst  
formats:T_HEADER  
varlist:<CurDate>  
phrasestr:<UGCListZ>  
phrasestr:DAILY RIVER FORECASTS  
phrasestr:NATIONAL WETHER SERVICE LOUISVILLE KY  
phrasestr:<CurDate>  
#
```

This is pretty straightforward. The list of valid template keywords is found in the RiverPro Reference Manual starting on page A-1. Let's look at the lines in this example:

```
name:rva_daily_fcst
```

All templates must have a unique name that does not contain spaces. Be sure to use a descriptive name that is easy to remember.

```
formats:T_HEADER  
varlist:<CurDate>
```

This pair lets the program know that you are going to use variable substitution in this section. The list of variables is found in the in the RiverPro Reference Manual starting on Page B-3. The TIME variable, <CurDate> holds the current date and time. I want the format of the variable to be T_HEADER, which looks like *900 AM EDT THU JUL 29 2001*. It is a good idea to specify the format for every variable. The list of valid formats is found in the RiverPro Reference Manual starting on page A-4. Notice that the `formats:/varlist:` pair produce no output on their own.

```
phrasestr:<UGCListZ>
```

This line inserts the UGC zone numbers for all the points that are printed.

```
INZ089>090-KYZ035-040-049-091400-
```

```
phrasestr:DAILY RIVER FORECASTS  
phrasestr:NATIONAL WEATHER SERVICE LOUISVILLE KY  
phrasestr:<CurDate>
```

These lines show what the header should look like. Notice the old AFOS "TTAA00" line is gone. The last line shows where we want the variable substitution to appear. Again, here is the output from this template:

```
INZ089>090-KYZ035-040-049-091400-  
DAILY RIVER FORECASTS  
NATIONAL WEATHER SERVICE LOUISVILLE KY  
1130 AM EDT FRI OCT 08 2001
```

This is all you need in the HEADER.TPL file. Now, lets tackle the TABULAR.TPL file. Here is the complete template:

```
# TABULAR SECTION TEMPLATES  
#  
name:rva_daily_fcst  
grpname:skip  
formats:"LOCATION" X12 "FS" X4 "OBS" X4 T_CAMDD X2 T_CAMDD X2 T_CAMDD  
varlist:                                     <Day1>      <Day2>      <Day3>  
miscwrt:  
formats:S18      F2.0 X4 F4.1 X4  F4.1 X4      F4.1 X4      F4.1  
varlist:<IdName> <FldStg> <ObsStg> <SpecFcstStg> <SpecFcstStg> <SpecFcstStg>  
spectime:TODAY 12:00      +1 0 1      +2 0 1      +3 0 1  
#OHIO RIVER  
fp_id:MKLK2
```

```

fp_id:MLUK2
fp_id:MLPK2
fp_id:CNNI3
fp_id:TELI3
#KENTUCKY RIVER
fp_id:FODK2
fp_id:HIBK2
fp_id:FFTK2
literal:
literal:END
#

```

Let's break it down:

```
name:rva_daily_fcst
```

As before, all templates must have a unique name within the same file. Since this template is only for the RVA_DAILY_FCST product, I gave it the same name as the template in the HEADER.TPL file. This makes it easy to connect products and templates.

```
grpname:skip
```

This puts a blank line in the product anytime a forecast group name is written. I will show you how this option will be useful later.

```

formats:"LOCATION" X12 "FS" X4 "OBS" X4 T_CAMDD X2 T_CAMDD X2 T_CAMDD
varlist:                <Day1>      <Day2>      <Day3>
miscwrt:

```

These lines make a header for the daily forecasts. The first line indicates the formats in the line. The "LOCATION" item puts the string of characters within the quotes on the line. The X12 format puts 12 spaces in the line. I could also have written " ", but the X format makes it easy to experiment with location placing. The T_CAMDD item shows the format for the variables <Day1>, <Day2>, <Day3> which are TIME variables. The T_CAMDD format indicates that I want month spelled out but abbreviated (CAM) followed by the date (DD). Notice how I placed the variables under their formats. This makes it easy to see what format is connected to which variable. The miscwrt: line causes the line to be written out. The formats: lines must be followed by either a miscwrt: or fp_id: line. Here is the result:

```
LOCATION                FS      OBS      OCT 09      OCT 10      OCT 11
```

Let's look at the next lines:

```

formats:S18          F2.0 X4  F4.1 X4  F4.1 X4          F4.1 X4          F4.1
varlist:<IdName> <FldStg> <ObsStg> <SpecFcstStg> <SpecFcstStg> <SpecFcstStg>
spectime:TODAY 12:00          +1 0 1          +2 0 1          +3 0 1

```

Here we are using many variables and have a format for each one. The first is the location name, <IDName>, which is a STRING. The S18 format reserves space for a string up to 18 spaces. If the string is less than 18 characters, blanks are added to the right. The next variable is the flood stage, <FldStg>, which is a FLOAT. The F2.0 reserves 2 spaces with no digits after the decimal point (and leaves off the decimal point as well).

The next variable is the observed stage, <ObsStg>, which is also a FLOAT. The F4.1 reserves 4 spaces with one digit after the decimal point. Notice, that if you want the decimal point printed, you must reserve a space for it.

The next three variables are specific forecast stages, <SpecFcstStg>, which are also FLOATS. The format is the same as <ObsStg>. However, we must tell RiverPro what forecast stages we want. That is what the next line, spectime: does. This line starts out with the base time, TODAY 12:00. Notice that the time is in UTC. The next three values tell what forecast we want. The first value is the number of days away from the base time, and can be negative or positive. In this example we have TODAY+1, which is of course the next day. The second tells how many hours away from the base time we want the forecast. Here we have 12:00+0 or 1200 UTC. The last number tells how big the window size is, so we will except any forecast from 1100 to 1300. This will cover daylight saving time shifts. Since we have three <SpecFcstStg> variables, we need three sets of numbers. Here is what these three lines will generate:

```
MARKLAND LOWER      51      12.1      12.5      12.3      12.3
```

We are almost done. However, we need to tell RiverPro what locations we want to print out. Here is the part of the template that does the trick:

```
#OHIO RIVER
fp_id:MKLN2
fp_id:MLUK2
fp_id:MLPK2
fp_id:CNNI3
fp_id:TELI3
#KENTUCKY RIVER
fp_id:FODK2
fp_id:HIBK2
fp_id:FFTK2
```

First, I have inserted a comment line to identify the river these forecast points are on. Comments can be used anywhere you want to, just start the line with a #; RiverPro ignores them. Next, every location you want to include should be included in a fp_id: line. Every time RiverPro encounters a location in a different forecast group it will output the group name. The fp_id:MKLN2 line will automatically cause the forecast group name, Ohio River, to be printed because it is the first in a group. The fp_id:FODK2 will cause the next forecast group name, Kentucky River, to be printed because FODK2 is the first in a new forecast group. Since we have grpname:skip line, RiverPro will put a blank line before each forecast group name. The result is this:

```
OHIO RIVER
MARKLAND LOWER      51      12.1      12.5      12.3      12.3
MCALPINE UPPER      23      12.2      12.2      12.2      12.2
MCALPINE LOWER      55       9.6       9.8       9.7       9.7
CANNELTON LOWER     42      10.9      10.9      10.9      10.9

KENTUCKY RIVER
FORD                 26       9.4       9.1       9.1       9.1
HIGH BRIDGE         30       8.3       8.9       8.9       8.9
FRANKFORT           31       7.2       6.8       6.8       6.8
```

Sometimes this automatic inclusion of the group name can cause problems. To turn this feature off use `grpname:off`.

Notice that I have included a forecast point, TELI3, which does not appear in the product. I only receive a forecast for this point whenever there is high water. Normally it does not appear, but it needs to be included in the template for the times it might appear. When issuing the product, if I do not select TELI3, it will not appear in the final product.

Finally, we want to put an ending line on the product. This is done by using the rest of the template:

```
literal:  
literal:END  
#
```

This command simply puts out exactly what you tell it. The line `literal:` with a space after it inserts a blank line in the product. The result of the final line is:

END

The template is complete. However, to get this product to run you must tell RiverPro how to use the template. First, you must edit the template into the HEADER.TPL and TABULAR.TPL file. The easiest way to do this is to click on **Settings/Modify product sections**. Now click on **Product Header /Edit Template File**. Do the same for the Tabular file.

After editing, your templates, you need to set up a PCC file. Select an unused product in the product creation window, and click on **Settings/Modify product sections**. Click on the **Product Header** button. Now click on the template you want and then **OK**. Do the same for the **Tabular** button. Make sure **Tabular** is selected as **1st** and all other sections are off. Click on **OK**. This selects the correct templates for the product.

Now, let's select the locations for the product. Click on **Settings/Select forecast points to include**, then click on the locations you want. Even if you have the id included in the template, you must select here to make it appear in the generated product. Click on **Close**.

Finally, let's lock in your selections into a PCC file. Click on **Settings/Save to settings file**. Ignore the top box and fill in the information in the bottom box. Here is what I used: **Filename:rva_daily_fcst.pcc**, **Product Id:KLMKRVASDF**, **Category:RVA**, **Description:RVA - Daily River Forecast**. Be sure to click on **Save Instructions for Including Specific Forecast Points**. If you do not do this, you will have to select the forecast points every time you generate this product. Click on **OK**.

Test your work by clicking on the product description and then **File/Create**. You should see your work. If your text is not lined up, just edit the template and create the product again. The interactive nature of RiverPro makes this process quick and easy.

A Sample Daily River Observation Product

Let's generate a product which contains data which does not have an explicit template variable for the data we want. To insert this type of data, we use the observed data template variable. You can see some examples of using the variable on the WHFS web page by Bill Wilson: www.nws.noaa.gov/oh/hod_whfs/pe_ver2rvrp.html.

Here is the new product we want to generate:

*OBSERVATIONS IN THE LOUISVILLE HYDROLOGIC AREA
NATIONAL WEATHER SERVICE LOUISVILLE KY
930 AM EDT FRI OCT 08 2001*

<i>LOCATION</i>	<i>FS</i>	<i>PRECIP</i>	<i>POOL/ STAGE</i>	<i>24-HR CHANGE</i>	<i>TAIL- WATER</i>	<i>24-HR CHANGE</i>	<i>WATER TEMP</i>
<i>OHIO RIVER</i>							
<i>MARKLAND LOCK</i>	<i>51</i>	<i>0.00</i>	<i>12.70</i>	<i>0.00</i>	<i>13.30</i>	<i>1.20</i>	<i>86</i>
<i>MCALPINE LOCK</i>	<i>55</i>	<i>0.00</i>	<i>12.70</i>	<i>0.40</i>	<i>10.10</i>	<i>0.30</i>	
<i>CANNELTON LOCK</i>	<i>42</i>	<i>0.00</i>	<i>9.70</i>	<i>-0.20</i>	<i>10.10</i>	<i>-0.40</i>	<i>82</i>
<i>OTHER OHIO RIVER GAGES</i>							
<i>CLIFTY CREEK</i>	<i>451</i>	<i>0.00</i>	<i>425.12</i>	<i>0.10</i>			
<i>KOSMOSDALE</i>			<i>10.12</i>	<i>-0.12</i>			

THIS DATA WAS FURNISHED BY THE COOPERATION OF THE NATIONAL WEATHER SERVICE...THE CORPS OF ENGINEERS...THE U.S. GEOLOGICAL SURVEY...AND THE KENTUCKY DIVISION OF WATER.

I call this product `rva_daily_river`. Here is what the template in the `HEADER.TPL` looks like:

```
# HEADER SECTION TEMPLATES
#
name:rva_daily_river
formats:T_HEADER
varlist:<CurDate>
phrasestr:OBSERVATIONS IN THE LOUISVILLE HYDROLOGIC AREA
phrasestr:NATIONAL WEATHER SERVICE LOUISVILLE KY
phrasestr:<CurDate>
#
```

As you can see, all header templates are similar. Review the previous example to explain the lines in this header template. Again, here is output from this template:

*OBSERVATIONS IN THE LOUISVILLE HYDROLOGIC AREA
NATIONAL WEATHER SERVICE LOUISVILLE KY
930 AM EDT FRI OCT 08 2001*

This is all you need in the `HEADER.TPL` file. Now let's tackle the `TABULAR.TPL` file. Here is the complete template:

```
# TABULAR SECTION TEMPLATES
```

```

#
name:rva_daily_rvr
grpname:noskip
formats:"LOCATION" X9 "FS" X4 "PRECIP" X3 "POOL/" X2 "24-HR" X4 "TAIL-" X2 &
      "24-HR" X3 "WATER"
miscwrt:
formats:X32 "STAGE" X2 "CHANGE" X3 "WATER" X2 "CHANGE" X2 "TEMP"
miscwrt:
formats:S16 X1 F2.0 X4 F5.2 X4 F5.2 X2 F6.2 X4 F5.2 X2 F6.2 X3 F2.0
varlist:<IdName> <FldStg> <PP,5004,RZ,Z,0|12:00|1> <HP,0,RZ,Z,0|12:00|1> &
      <HP,0,RZ,Z,0|12:00|1,CHG24> <HT,0,RZ,Z,0|12:00|1> &
<HT,0,RZ,Z,0|12:00|1,CHG24> <TW,0,RZ,Z,0|12:00|1>
locid:MKLK2
locid:MLPK2
locid:CNNI3
literal:
literal:OTHER OHIO RIVER GAGES
formats:S15 X1 F3.0 X4 F5.2 X3 F6.2 X2 F7.2
varlist:<IdName> <FldStg> <PP,5004,RZ,Z,0|12:00|1> <HG,0,RZ,Z,0|12:00|2> &
      <HG,0,RG,Z,0|12:00|2,CHG24>
locid:CLFI3
formats:S15 X8 F5.2 X2 F6.2
varlist:<IdName> <HG,0,RG,Z,0|12:00|1> <HG,0,RG,Z,0|12:00|1,CHG24>
locid:KMDK2
literal:
literal:THIS DATA WAS FURNISHED BY THE COOPERATION OF THE NATIONAL WEATHER
literal:SERVICE...THE CORPS OF ENGINEERS...THE U.S. GEOLOGICAL SURVEY...AND
literal:THE KENTUCKY DIVISION OF WATER.

```

While this looks quite complicated, it is really straightforward. Let's look at the first two lines:

```

name:rva_daily_river
grpname:noskip

```

Every template needs a name, and I used one that tells me the product in which this template is used. The next line instructs RiverPro not to skip a line every time a new forecast group name is printed.

The next 6 lines create a header to the product:

```

formats:"LOCATION" X9 "FS" X4 "PRECIP" X3 "POOL/" X2 "24-HR" X4 "TAIL-" X2 &
      "24-HR" X3 "WATER"
miscwrt:
formats:X32 "STAGE" X2 "CHANGE" X3 "WATER" X2 "CHANGE" X2 "TEMP"
miscwrt:
literal:

```

Note that when a line gets too long, you must put an ampersand (&) at the end of it. I could have used `literal:` lines since I have no variables to insert in the `formats:` lines.

Here is the header that these lines produce:

```

LOCATION          FS      PRECIP  POOL/  24-HR   TAIL-  24-HR   WATER
                STAGE  CHANGE  WATER  CHANGE  TEMP

```

Notice how I am going to put both pool data (HP) and stage data (HG) in the same column.

The next lines show how to include data which do not have an explicit template variable:

```
formats:S16 X1 F2.0 X4 F5.2 X4 F5.2 X2 F6.2 X4 F5.2 X2 F6.2 X3 F2.0
varlist:<IdName> <FldStg> <PP,5004,RZ,Z,0|12:00|1> <HP,0,RZ,Z,0|12:00|1> &
      <HP,0,RZ,Z,0|12:00|1,CHG24> <HT,0,RZ,Z,0|12:00|1> &
      <HT,0,RZ,Z,0|12:00|1,CHG24> <TW,0,RZ,Z,0|12:00|1>
```

Remember, `formats:` and `varlist:` lines form pairs. There is nothing new in the `formats:` line or the first two variables in the `varlist:` line. However, the third variable, `<PP,5004,RZ,Z,0|12:00|1>`, is an observed data template variable and needs some explanation.

The SHEF descriptor for a piece of data looks like this, `PEDTSE`, where `PE` is the Physical Element code, `D` is the Duration code, `TS` is the Type and Source code, and `E` is the Extremum code. Thus, daily observer precipitation ending at 1200 UTC would be coded `DH12/PPRZZ`. The RiverPro template variable for this data would be `<PP,5004,RZ,Z,0|12:00|1>`.

In this variable, the first two letters (`PP`) are physical element, while the value (5004) is the duration code as explained in Note 2 of Appendix H-2 in the SHEF Manual and on page 38 of the RiverPro Reference Manual. The most common duration codes and their numbers are: instantaneous - 0, hourly - 1001, daily - 2001, and since 7 AM - 5004.

The next two letters (`RZ`) in the above example are the Type and Source. The most common are: unspecified - `RZ`, satellite - `RG`, phone - `RP`, and radio - `RR`. The next letter (`Z`) is the Extremum. The most common are: maximum of day - `X`, minimum of day - `N`, 6 hour maximum - `R`, 6 hour minimum - `H`, and other - `Z`.

The last item in this variable (`0|12:00|1`) shows the time of the observation desired. It is represented like this: `reference_day|hour:minute|hour_window`. The `reference_day` tells us how many days away from today we want the observation. Today is 0, yesterday is -1, and so forth. The `hour:minute` is the time of the data we want using UTC. The `hour_window` is the number of hours on either side of `hour:minute` to accept data. Thus, `0|12:00|1` would represent the data at 1200 UTC today with an acceptance window from 1100-1300 UTC. To get the latest data, just use the phrase `LATEST`. Notice that the `<ObsStg>` template variable for an observer river stage could be coded `<HG,0,RZ,Z,LATEST>`.

With this information, we can see that `<PP,5004,RZ,Z,0|12:00|2>` would retrieve the daily precipitation ending at 1200 UTC or `PPRZZ` or just `PPP`. `<HP,0,RZ,Z,0|12:00|2>` would retrieve the pool stage at 1200 UTC or `HPIRZZ` or just `HP`. Both variables would contain an acceptance window from 1000-1400 UTC. I realize that the plus side of the acceptance window could cause some problems with the `PPP` data.

The next variable in our example, `<HP,0,RZ,Z,0|12:00|1,CHG24>`, shows an option in the template variable. The `CHG24` calculates the 24-hour change in the data. The possible values for this element are: `CHG##` - calculate the change, `MAX##` - calculate the maximum, `MIN##` - calculate the minimum, and `ACC` - accumulate precipitation. `##` can be any number of hours from 0 to 370. I will explain the `ACC` option later.

Another option is `TIME`, which is used to display the time of the data and not the actual value. Here is an example: we want to display the latest stage of a satellite gage, the time of the reading, the highest reading in the

last 24 hours, and the time of the maximum stage. We would use the following `varlist` :

```
varlist:<HG,0, RG, Z, LATEST> <HG,0, RG, Z, LATEST, TIME> &  
      <HG,0, RG, Z, LATEST, MAX24> <HG,0, RG, Z, LATEST, MAX24, TIME>
```

With this knowledge, template variables are easy to understand. `<HT,0, RZ, Z, 0|12:00|1>` is the tailwater reading at 1200 UTC, `<HT,0, RZ, Z, 0|12:00|1, CHG24>` is the 24 hour change, and `<TW,0, RZ, Z, 0|12:00|1>` is the water temperature.

The next three lines tell RiverPro to get the desired data for three locations :

```
locid:MCLK2  
locid:MLPK2  
locid:CNNI3
```

We are using `locid` : instead of `fp_id`. By doing this, we can one get data from non-forecast points as well as forecast points. The amount of template variables available using `locid` : is limited. The only variables that are allowed are `<Id>`, `<IdName>`, `<County>`, `<StateId>`, `<StateName>`, `<River>`, `<BankStg>`, `<WStg>`, `<FldStg>`, `<FldFlow>`, `<ZDatum>`, `<Reach>`, and `<Proximity>`. If you do need to include information about a non-forecast point which is not included in the template variables, you must hard code the information in the `formats` : line. However, there is an advantage to using `locid` : even for forecast points. You do not have to preselect the forecast point you want included in a product in the PCC file if you use `locid` : rather than `fp_id` : . I will show how this works a little later.

These lines produce the following output:

```
OHIO RIVER  
MARKLAND LOCK      51      0.00   12.70   0.00   13.30   1.20      86  
MCALPINE LOCK      55      0.00   12.70   0.40   10.10   0.30      86  
CANNELTON LOCK     42      0.00    9.70  -0.20   10.10  -0.40      82
```

Notice that McAlpine Lock does not have a water temperature. Normally, RiverPro would put MSG (missing) there but I changed it to a blank. This is performed in HydroBase/Setup/RiverPro General Parameters/ String to Use for Missing/Data Value. While you are there, check out the other parameters you can set. I could have also included in the template:

```
msgdata :
```

Which would have changed the value for missing just for this product. If the line is `msgdata:skip`, then RiverPro would skip that line if any of the data for that station in that line is missing.

The next 6 lines in our example template show how to display similar information for a gage on the river which is not a lock. Because the physical element is different (no HP/HT just HG) it has to have a new `varlist` : Also the flood stage at Clifty Creek is three digits wide so I must also specify a new `formats` : line as well. If the flood stage at Clifty Creek were two digits wide like the above locations I could have used the same `formats` : line. RiverPro will let two `varlist` : lines share the same `formats` : line. When RiverPro finds a new `varlist` : line, it uses the most recent `formats` : line. If you are sharing `formats` : lines, make sure that the number of `formats` is the same as the number of variables in both `varlist` : lines.

```
literal:
literal:OTHER OHIO RIVER GAGES
formats:S15 X1 F3.0 X4 F5.2 X3 F6.2 X2 F7.2
varlist:<IdName> <FldStg> <PP,5004,RZ,Z,0|12:00|1> <HG,0,RZ,Z,0|12:00|2> &
      <HG,0,RG,Z,0|12:00|2,CHG24>
locid:CLFI3
```

This gives the following output:

```
OTHER OHIO RIVER GAGES
CLIFTY CREEK      451          0.00  425.12   0.10
```

Here are the lines to print out the information for Kosmosdale, a another gage which is not a lock on the Ohio River and which is not a forecast point:

```
formats:S15 X8 X22 F5.2 X2 F6.2
varlist:<IdName> <KMDK2,HG,0,RG,Z,0|12:00|1> <HG,0,RG,Z,0|12:00|1,CHG24>
locid:
```

Notice that I did not include <FldStg> in the varlist: because Kosmosdale does not have a flood stage. These lines produce the following output:

```
KOSMOSDALE                10.12  -0.12
```

Lastly, I like to give the NWS, USCE, USGS, and Kentucky Division of Water a little free advertising. I use the following literal: lines:

```
literal:
literal:THIS DATA WAS FURNISHED BY THE COOPERATION OF THE NATIONAL WEATHER
literal:SERVICE...THE CORPS OF ENGINEERS...THE U.S. GEOLOGICAL SURVEY...AND
literal:THE KENTUCKY DIVISION OF WATER.
```

This produces the last lines of the product:

```
THIS DATA WAS FURNISHED BY THE COOPERATION OF THE NATIONAL WEATHER
SERVICE...THE CORPS OF ENGINEERS...THE U.S. GEOLOGICAL SURVEY...AND
THE KENTUCKY DIVISION OF WATER.
```

That is all for this template. Review the first example in order to see how to set up a product control file (.PCC) using these templates, and how to generate the product. When you generate a product using a template that contains no fp_id: commands, be sure to first clear all the selected forecasts by clicking on Settings/Select forecast points to include, Clear, and Close. Then save the PCC file with the Save Instructions for Including Specific Forecast Points checked. If you do not do this and forecast points are accidentally selected before you create the product, you will see the forecast points tacked on the end of your product.

A Sample Daily Product Using Multiple Columns

If you want to create a product where you have multiple locations on the same line you cannot use either `fp_id:` or `locid:`. You must hard code the information about the location into the formats line, the location id into observed data template variables, and follow the lines with a `miscwrt:` line. Using this technique, every line output in the product will require three lines in the template. Also, there is no way to prevent the location name from appearing if the data is missing.

Lets look at 24-hour precipitation. Precipitation comes from basically three types of gages. The first is manually read and is reported either as **PPD** or **PPP**. To extract precipitation from this type of gage, just convert the SHEF code to template variable. For example, **PPPRZZ** for Aberdeen, KY would be encoded as `<ABEK2, PP, 5004, RZ, Z, 0|12:00|2>` and **PPDRZZ** as `<ABEK2, PP, 2001, RZ, Z, 0|12:00|2>`.

The second type is an automatic gage that reports a precipitation accumulator, which is encoded as **PC**. Both LARC and DCP precipitation gages usually work this way. To get the 24-hour precipitation from the LARC at Paris, KY, you would encode: `<PRSK2, PC, 2001, RP, Z, 0|12:00|2>`.

The last type of gage is one that reports hourly precipitation continuously as **PPH**. An example is an IFLOWS gage or the ASOS precipitation data. You must tell RiverPro to add up the individual values in order to come up with the 24-hour total by using the **ACC** option. To get the 24-hour precipitation from IFLOWS gage at Berry, KY you would encode: `<BRYK2, PP, 2001, RR, Z, 0|12:00|2, ACC>`.

Putting all this information together, produces the following sample of a daily precipitation template:

```
literal:KENTUCKY COOPERATIVE OBSERVERS...
formats:"ABERDEEN      " F5.2 " PARKSVILLE      " F5.2 " WOODBURY      " F5.2
varlist:<ABEK2, PP, 5004, RZ, Z, 0|12:00|2> <PRKK2, PP, 5004, RZ, Z, 0|12:00|2> &
        <WDBK2, PP, 2001, RZ, Z, 0|12:00|2>
miscwrt:
literal:
literal:KENTUCKY AUTOMATED GAGES...
formats:"ALBANY        " F5.2 " BERRY          " F5.2 " PARIS          " F5.2
varlist:<ALBK2, PC, 2001, RG, Z, 0|12:00|2> <BRYK2, PP, 2001, RR, Z, 0|12:00|2, ACC> &
        <PRSK2, PC, 2001, RP, Z, 0|12:00|2>
miscwrt:
```

The output is:

```
KENTUCKY COOPERATIVE OBSERVERS...
ABERDEEN      0.00 PARKSVILLE      WOODBURY      0.00

KENTUCKY AUTOMATED GAGES...
ALBANY        BERRY          0.00 PARIS          0.00
```

Last, let's look at temperatures. Most cooperative observers with temperature gages send in three temperatures a day, including maximum temperature encoded as **TX**, minimum temperature encoded as **TN**, and current temperature encoded as **TA**. However, if you look in HydroView, you will not find **TX** and **TN** as physical elements, only **TA**. It turns out that **TX** is converted to **TAIRZX** and **TN** is converted to **TAIRZN**. So the template variable for maximum would be `<TA, 0, RZ, X, 0|12:00|1>`, while the minimum would be `<TA, 0, RZ, N, 0|12:00|1>`. ASOS data is different. When ASOS sends **TAIRZX**, it is based on calendar

day data, not observer day (7AM) data. But, here is a template variable to tell RiverPro to find the highest 6 hour temperature during the last 18 hours at the Frankfort ASOS:

<FFT,TA,0,RZ,R,0|12:00|1,MAX18>. Notice that you want to use 18 hours which will look at the last four 6 hour reports, rather than 24 which would look at the last five six hour reports. Actually, any number between 18 and 23 hours could be used. For the minimum temperature use:

<FFT,TA,0,RZ,H,0|12:00|1,MIN18>. Here is a section of a template that outputs observer and ASOS temperature data:

```
literal:LOCATION                MAXIMUM        MINIMUM        OBSERVED
formats:"BRANDENBERG          " X8 F2.0 X11 F2.0 X11 F2.0
varlist:<BNBK2,TA,0,RZ,X,0|12:00|1> <BNBK2,TA,0,RZ,N,0|12:00|1> &
        <BNBK2,TA,0,RZ,Z,0|12:00|1>
miscwrt:
formats:"FRANKFORT            " X8 F2.0 X11 F2.0 X11 F2.0
varlist:<FFT,TA,0,RZ,R,0|12:00|1,MAX18> <FFT,TA,0,RZ,H,0|12:00|1,MIN18>
        <FFT,TA,0,RZ,Z,0|12:00|1>
miscwrt:
formats:"ROCHESTER            " X8 F2.0 X11 F2.0 X11 F2.0
varlist:<RCHK2,TA,0,RZ,X,0|12:00|1> <RCHK2,TA,0,RZ,N,0|12:00|1> &
        <RCHK2,TA,0,RZ,Z,0|12:00|1>
```

Here is the output:

<i>LOCATION</i>	<i>MAXIMUM</i>	<i>MINIMUM</i>	<i>OBSERVED</i>
<i>BRANDENBERG</i>	<i>80</i>	<i>54</i>	<i>56</i>
<i>FRANKFORT</i>	<i>82</i>	<i>55</i>	<i>55</i>
<i>ROCHESTER</i>	<i>78</i>	<i>51</i>	<i>51</i>

In a related problem, one needs to make a decision when exacting crest data using template variables. Here are the two choices: <HG,I,RZ,X,LATEST> or <HG,I,RZ,Z,LATEST,MAX24>. These could refer to different data. If the crest was encoded as HX, which gets converted to HGIRZX, you would use the first choice. However, the second choice scans the database and extracts the highest value, even if it was not encoded as a crest. The disadvantage with the second choice is that it will report a value, even if it is not a true crest, while the first will not.

A Sample SHEF Encoded State Temperature/Precipitation Product

Using the techniques we learned above, we can see how easy it is to use RiverPro to create an STP that is SHEF encoded. Here is the complete template:

```
name:stp_am
formats:".B LMK " T_MMDDS " E DH00/TAIRZX/DH08/TAIRZP/PPDRZZ"
varlist:<Day0>
miscwrt:
literal::
literal:: HIGHEST TEMPERATURE YESTERDAY.
literal:: LOWEST TEMPERATURE SINCE 8PM EDT LAST NIGHT.
literal:: PRECIPITATION FOR THE 24 HOURS ENDING AT 8AM EDT TODAY.
literal::
literal::...KENTUCKY...
literal::
          HI      LO      PCPN
formats:S3 " : " S14 " : " F3.0 " / " F3.0 " / " F5.2
varlist:<Id> <IdName> <TA,0,RZ,X,0|05:00|1> <TA,0,RZ,H,0|12:00|1,MIN6> &
        <PP,1024,RZ,Z,0|12:00|1,ACC>
locid:CVG
locid:JKL
locid:LEX
locid:PAH
locid:HOP
locid:FTK
locid:BWG
locid:SDF
locid:LMK
locid:FFT
locid:LOZ
literal:.END
#
```

And here is the output:

```
.B LMK 0901 E DH00/TAIRZX/DH08/TAIRZP/PPDRZZ
:
: HIGHEST TEMPERATURE YESTERDAY.
: LOWEST TEMPERATURE SINCE 8PM EDT LAST NIGHT.
: PRECIPITATION FOR THE 24 HOURS ENDING AT 8 AM EDT TODAY.
:
:...KENTUCKY...
:
          HI      LO      PCPN
CVG : CINCINNATI      : 90 / 71 / 0.00
JKL : JACKSON         : 85 / 70 / 0.00
LEX : LEXINGTON       : 89 / 70 / 0.01
PAH : PADUCAH         : 92 / 66 / 0.00
HOP : FORT CAMPBELL  :   /   / 0.00
FTK : FORT KNOX      :   /   / 0.00
BWG : BOWLING GREEN  : 88 / 71 / 0.00
SDF : LOUISVILLE APT : 90 / 74 / T
LMK : LOUISVILLE NWS : 88 / 72 / 0.00
FFT : FRANKFORT      : 88 /   / 0.03
```

```
LOZ : LONDON          : 87 / 67 / 0.00
.END
```

There is very little that is new here. Lets look at each section. The SHEF header and comments are created by this section of the template:

```
name:stp_am
formats:".B LMK " T_MMDDS " E DH00/TAIRZX/DH08/TAIRZP/PPDRZZ"
varlist:<Day0>
miscwrt:
literal::
literal:: HIGHEST TEMPERATURE YESTERDAY.
literal:: LOWEST TEMPERATURE SINCE 8PM EDT LAST NIGHT.
literal:: PRECIPITATION FOR THE 24 HOURS ENDING AT 8AM EDT TODAY.
literal::
literal::...KENTUCKY...
literal::                HI      LO      PCPN
```

We are using the <Day0> template variable with a format of T_MMDDS to create the proper SHEF date code:

```
.B LMK 0901 E DH00/TAIRZX/DH08/TAIRZP/PPDRZZ
:
: HIGHEST TEMPERATURE YESTERDAY.
: LOWEST TEMPERATURE SINCE 8PM EDT LAST NIGHT.
: PRECIPITATION FOR THE 24 HOURS ENDING AT 8 AM EDT TODAY.
:
: ...KENTUCKY...
:                HI      LO      PCPN
```

The next section shows how to retrieve the required data for the airports:

```
formats:S3 " : " S14 " : " F3.0 " / " F3.0 " / " F5.2
varlist:<Id> <IdName> <TA,0,RZ,X,0|05:00|1> <TA,0,RZ,H,0|12:00|1,MIN6> &
      <PP,1024,RZ,Z,0|12:00|1,ACC>
locid:CVG
locid:JKL
locid:LEX
locid:PAH
locid:HOP
locid:FTK
locid:BWG
locid:SDF
locid:LMK
locid:FFT
locid:LOZ
literal:.END
```

We want the high temperature for yesterday and the SHEF code is E DH00/TAIRZX, which get converted to <TA,0,RZ,X,0|05:00|1> as an observed data template variable in EDT. The low temperature is the lowest in the last 12 hours, which is DH08/TAIRZP in SHEF and <TA,0,RZ,H,0|12:00|1,MIN6> as an template variable. The precipitation is for the last 24 hours which is PPDRZZ in SHEF, and <PP, 1024, RZ, Z, 0|12:00|1, ACC> as a template variable.

This give us the output we want:

```
CVG : CINCINNATI      : 90 / 71 / 0.00
JKL : JACKSON         : 85 / 70 / 0.00
LEX : LEXINGTON       : 89 / 70 / 0.01
PAH : PADUCAH        : 92 / 66 / 0.00
HOP : FORT CAMPBELL  :    /    / 0.00
FTK : FORT KNOX      :    /    / 0.00
BWG : BOWLING GREEN  : 88 / 71 / 0.00
SDF : LOUISVILLE APT : 90 / 74 / T
LMK : LOUISVILLE NWS : 88 / 72 / 0.00
FFT : FRANKFORT      : 88 /    / 0.03
LOZ : LONDON         : 87 / 67 / 0.00
.END
```

If you find that you cannot extract data out of your database, be sure to look at how the data was stored using HydroView. You can learn a lot about the finer points of SHEF by simply looking at your data. If your observed data is in HydroView, you should be able to come up with a template variable to display it in RiverPro.

A Sample SHEF Encoded Regional Temperature/Precipitation Product using Cooperative Data

As we saw in the previous example, RiverPro can be used to generate SHEF encoded reports. However, with cooperative data, this can lead to a problem. Thanks to Bill Morris of Chicago for pointing out the problem and finding out a simple solution. Assume that we are Chicago, and that the cooperative station at MOSI2 made a report on 06/10 at 6:30 CST. We have a RiverPro template that generates this product:

```
.B CHI 0610 C DH07/TAIRZX/TAIRZN/PPDRZZ
MOSI2 : MORRIS          : 95 / 70 / 0.13
.END
```

Unfortunately, the SHEF decoder in AWIPS will pick up this product and create a phantom observation for MOSI2 at 7:00 CST along with the original one at 6:30. We need to be sure that the time of the observation in the generated product exactly agrees with the original observation. The way to do this is to include the actual time of the observation along with the observation. Our correct report will look like this:

```
.B CHI 0610 C DH07/TAIRZX/TAIRZN/PPDRZZ
MOSI2 : MORRIS          : DD100630 / 95 / 70 / 0.13
.END
```

The template variable `<PP,2001,RZ,Z,0|12:00|2,TIME>` will retrieve the time of the observation. This will work as long as the observer reports their precipitation as **PP** and reports a precipitation even on days when it does not rain. I will use the **DD** SHEF code along with the format `T_DDHHNN` because at this time, there is no format that would work with the **DH** SHEF code.

Here is the template that will do the job:

```
name:rtp_am
formats:".B CHI " T_MMDDS " C DH07/TAIRZX/TAIRZN/PPDRZZ
varlist:<Day0>
miscwrt:
literal::24 HOUR MAXIMUM AND MINIMUM TEMPERATURES AND PRECIPITATION
literal::ENDING AT THE INDICATED TIME THIS MORNING
literal::
literal::...ILLINOIS...
literal::
literal::
literal::          DATE/TIME    HI    LO    PCPN
formats:S5 " : " S14 " : "DD" T_DDHHNN " / " F3.0 " / " F3.0 " / " F5.2
varlist:<Id> <Idname> <PP,2001,RZ,Z,0|12:00|2,TIME> <TA,0,RZ,X,0|12:00|2> &
        <TA,0,RZ,N,0|12:00|2> <PP,2001,RZ,Z,0|12:00|2>
locid:ANTI2
locid:MOSI2
locid:PFDI2
literal:.END
```

Here is the output:

```
.B CHI 0610 C DH07/TAIRZX/TAIRZN/PPDRZZ
:24 HOUR MAXIMUM AND MINIMUM TEMPERATURE AND PRECIPITATION
:ENDING AT THE INDICATED TIME THIS MORNING
:
:...ILLINOIS...
```

```
:  
:  
      DATE/TIME    HI    LO    PCPN  
ANTI2 : ANTIOCH      : DD100700 / 94 / 68 / 0.00  
MOSI2 : MORRIS       : DD100630 / 95 / 70 / 0.13  
PFDI2 : PLAINFIELD  : DD100730 /    /    / 0.00  
.END
```

Notice that there are no problems with this template, even though Plainfield does not have thermometers. As long as they always report precipitation, the template will work.

A Daily River Summary (RVD) Product

The RVD for Louisville is tricky. Since the Ohio River is navigable, I must issue forecasts for both the upper and lower gages on the river, according to the wishes of the users. However, the product must also be SHEF encoded correctly. The sample template included in the original instructions for the RVD will not work here and we must create our own. Here is a section of the final product:

```
INZ089>92-KYZ025-028-030>031-035-040-049-110400-
DAILY RIVER SUMMARY
NATIONAL WEATHER SERVICE LOUISVILLE KY
1252 PM EDT TUE JUL 10 2001
```

```
.B LMK 0710 DH12/DC071015/HP/HT/DRH+24/HPIFF/HTIFF
.B1 DRH+48/HPIFF/HTIFF/DRH+72/HPIFF/HTIFF
:      STATION          FLOOD    7AM    24-HR          7AM
:ID    NAME             STAGE    STAGE  CHANGE          FORECASTS
:                                           WED     THU     FRI
:OHIO RIVER
MKLK2 :MARKLAND LOWER  51.0: / 22.7 /:  5.5: / 23.2// 18.9// 15.3
MLUK2 :MCALPINE UPPER  23.0:  12.7//:  0.1:  12.5// 12.5// 12.5/
.END
```

Here is the template for this product:

```
name:rvd
msgdata:M
formats:".B LMK ` T_U%m%d X1 T_UDC%m%d15 ` /DH12/HP/HT/DRH+24/HPIFF/HTIFF"
varlist:<Day0> <Day0>
miscwrt:
formats:".B1 DRH+48/HPIFF/HTIFF/DRH+72/HPIFF/HTIFF"
miscwrt:
literal::      STATION          FLOOD    7AM    24-HR          7AM
literal::ID    NAME             STAGE    STAGE  CHANGE          FORECASTS
formats:":` X47 T_U%a X4 T_U%a X4 T_U%a
varlist:<Day1> <Day2> <Day3>
miscwrt:
literal::OHIO RIVER
formats:S5 ` :` S16 F4.1 `: / ` F4.1 ` /:` F5.1 `: / ` F4.1 `// ` &
      F4.1 `// ` F4.1
varlist:<Id> <IdName> <FldStg> <HT,0,R*,Z,0|12:00|3> &
      <HT,0,RZ,Z,0|12:00|3,CHG24> <HT,0,FF,Z,1|12:00|3> <HT,0,FF,Z,2|12:00|3> &
      <HT,0,FF,Z,3|12:00|3>
fp_id:MKLK2
formats:S5 ` :` S16 F4.1 `:  ` F4.1 `//:` F5.1 `:  ` F4.1 `// ` &
      F4.1 `// ` F4.1 `/"
varlist:<Id> <Name> <FloodStg> <HP,0,R*,Z,0|12:00|3> &
<HP,0,RZ,Z,0|12:00|3,CHG24> <HP,0,FF,Z,1|12:00|3> <HP,0,FF,Z,2|12:00|3> &
<HP,0,FF,Z,3|12:00|3>
fp_id:MLUK2
literal:.END
```

There are some new tricks here. Look at this line in the product:

```
.B LMK 0710 DC071015/DH12/HP/HT/DRH+24/HPIFF/HTIFF
```

This is a forecast value SHEF header which requires a DC group with month, date, and hour of creation. The new easy way to do this is use the new T_Uxxx format. In order to create *DCmmd15* we use this format: T_UDC%m%d15. Lets look at this format in detail:

T_U - User defined format, DC - insert these letters, %m - insert the month as a decimal number, %d - insert the day of the month as a decimal number, 15 - insert these numbers. Thus, T_UDC%m%d15 generates *DC071115* for July 11 at 15:00Z.

Notice, that I hard coded the creation time hour as 15Z. I did this to get around a problem. If I would have used %H format, RiverPro would have inserted the local time, not UTC. This could lead to a creation time that was before the data time. By hard coding 15, I set the creation time always at 15Z, which is when the RVD is usually issued. Of course, I could have coded the .B line in local time too.

You can see that the T_Uxxx formats offers the ability to format a time in about any format you can imagine. Look in the RiverPro Manual for more information. By using the T_Uxxx format, you can also duplicate the most of the other T_ formats as well. Look at this line:

```
formats:":" X46 T_U%a X4 T_U%a X4 T_U%a
```

I could have encoded it like this:

```
formats:":" X46 T_AW X4 T_AW X4 T_AW
```

The resulting output will be the same:

```
:                                WED     THU     FRI
```

Some other new tricks show up in this line:

```
formats:S5 " : " S16 F4.1 " : / " F4.1 " / : " F5.1 " : / " F4.1 " // " &
          F4.1 " // " F4.1
varlist:<Id> <IdName> <FldStg> <HT,0,R*,Z,0|12:00|3> &
          <HT,0,RZ,Z,0|12:00|3,CHG24> <HT,0,FF,Z,1|12:00|3> <HT,0,FF,Z,2|12:00|3> &
          <HT,0,FF,Z,3|12:00|3>
```

On the Ohio River, I receive data from both an observer and a satellite gage. Most of the time I would like to use the observer reading. However, if the observer reading is missing, I would like the data from the satellite to show up. You can do this using a wildcard in the physical element part of the format, R*. The behavior of the wildcard is defined in HydroBase under **Data Ingest/Ingest Filter/TypeSource Rank:**. Be sure that you do NOT use a wildcard when calculating a change. If you do, RiverPro will hang. Notice, I use the wildcard for the observed data, <HT,0,R*,Z,0|12:00|3>, but I do not for the 24 hour change value, <HT,0,RZ,Z,0|12:00|3,CHG24>.

For forecast values, you can use the data template variable form and no longer have to use spectime: lines. Thus to get the 24 hour forecast value I can use <HT,0,FF,Z,1|12:00|3> instead of <SpecFcstStg> with a spectime: TODAY 12:00 +1 0 3. I find the new way to be easier, however both methods will produce the same line:

```
MKLN2 :MARKLAND LOWER 51.0: / 22.7 /: 5.5: / 23.2// 18.9// 15.3
```

As you can see, if all the forecast points used HG as their physical element, this preformat would be much simpler. But using the newer features of RiverPro, one can generate a correctly SHEF encoded RVD for a navigable river that is easier to read, as long as seemingly meaningless slashes don't bother you.

I hope these examples will allow you to use RiverPro to produce a product containing most of the data you desire. While at first glance the tabular template looks complicated, after working with it for awhile, it becomes easy to use. With each new release of RiverPro, the templates become shorter and easier to make. However, the old style templates still work.

When designing tabular products, it is important to make sure the columns line up. I have found the attached grid makes this process easier. Enjoy!

