US Army Corps
of Engineers®
Hydrologic Engineering Center

Hydrologic Modeling System HEC-HMS

Implementing the Program Interface in Alternate Languages

March 2007

Introduction

Version 3 of the Hydrologic Modeling System (HEC-HMS) was designed to operate independent of regional and language settings on the computer. The internal logic of the program does not make any assumptions about which language is used to display information to the user or receive input. Such tasks as number formatting, calendar dates, comparing the names of components, and sorting lists of components are carried out using appropriate local conventions. This method of designing the program is called *internationalization*. It means that the program can operate successfully in any country for which such conventions have been established.

The program is delivered with textual information in the English language. This information is used to fill the various parts of the interface such as window titles, labels, tooltips, and button names. It is also used to display note, warning, or error messages when necessary. The inclusion of textual information for a specific language is called *localization*. While HEC is not currently planning to provide localization for languages other than English, it is possible for users to develop their own localizations. This document describes how to perform localization of the textual information required by the program. All text in the program interface can be localized except for legal notices about the terms and conditions of usage, and certain contact information.

Locales

A *locale* is used to describe a specific local convention and includes both a county and a language. Some languages are only spoken in a single country. For example, Japanese is only associated with the country of Japan. Other languages are spoken in multiple countries. For example, Spanish is spoken in Spain and most of the countries in Central and South America as well. Some countries have multiple languages. For example, both English and French are spoken in Canada. In general, there is a locale for each official language in a country.

A locale is identified by two parts: the language first and the country second. Each part uses a two-letter code with the language code in lower case and the country code capitalized. For example, the code for English in the United States is "en_US". Language and country codes have been defined by the International Standards Organization. The definition of language codes is called ISO 639 and the definition of country codes is called ISO 3166.

Locales have been established for all countries and languages in the world. However, only some locales are compatible with the internationalization procedures used to develop the program. Appendix A lists the locales for which localization can be accomplished with a very high probability of success. Appendix B lists the locales for which localization should be successful, but problems may be encountered.

Interface Bundles

The textual information for the interface is organized into three files; each file is called a *resource bundle*. One file is used for textual information required for basin model component editors, global editors, and other editors or windows related to basin model components. A second file is used for information used for meteorologic model components. A third file holds information for the project, time-series data, paired data, grid data, and other remaining components. The default resource bundles are stored where the program is installed, for example:

```
C:\Program Files\HEC\HEC-HMS\3.0.0\ui
```

The files are in simple text format. Strictly speaking, the information in each file uses the UNICODE specification system. However, this is not obvious by looking at the files. The portion of the UNICODE specification necessary for the English alphabet matches the ASCII system which is much more recognizable. While the ASCII system is capable of displaying simple letters, it lacks the ability to display many non-English alphabets. One aspect missing from ASCII is diacritical marks such as the grave (for example dignità) used in several languages including Italian. ASCII also does not have the ability to display complex scripts required in writing systems such as Chinese or Devanagari. The UNICODE system contains a unique identifier for 96,477 different characters, symbols, and marks. Where possible, the UNICODE identifier is equivalent to ASCII. This allows the resource bundles to be very simple for the English localization, but able to use other letters, marks, or writing systems when necessary.

Each piece of information required for the interface is stored as one line in a resource bundle. The line begins with the identifier used by the program to locate a specific resource item. The text that should be used in place of the identifier follows the equal sign. Following are some sample lines from a resource bundle:

```
Element_BasinName_L=Basin Name:
Element_BasinName_TT=Basin Model Name
Element_Name_L=Element Name:
Element_Name_TT=Hydrologic Element Name
Element_Description_L=Description:
Element_Description_TT=Element Description
```

These example lines are used for hydrologic element component editors such as the subbasin element editor. The program will label the name of the basin model with whatever text follows the equal sign of the first resource item. Specifically, the program requests the identifier `Element_BasinName_L` from the resource bundle and `Basin Name:` is returned for display in the interface. The result is the component editor shown in Figure 1.

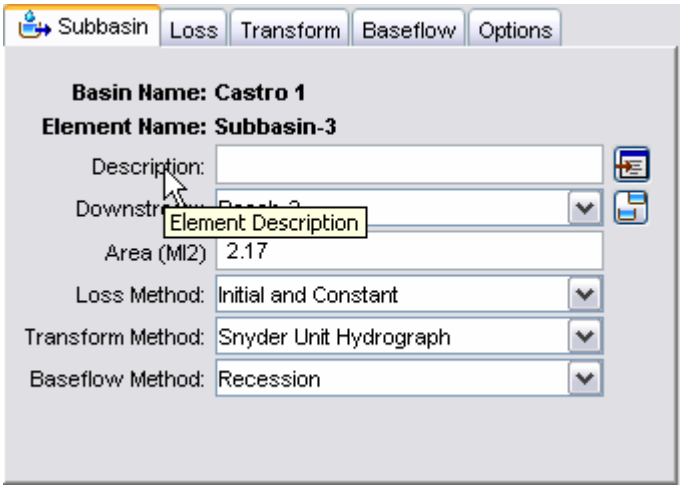

Figure 1. Subbasin component editor where the text has been loaded from the resource bundle. Notice the tooltip for the element description.

Several conventions are used throughout the resource bundles. The beginning of the identifier is generally the same for all textual information in a menu, component editor, or window. All of the resources for a particular component are stored together. A label usually ends with a capital letter "L" while the tooltip for the label ends with the letters "TT". Any units for a label are automatically added by the program after retrieving the

resource item from the resource bundle. Finally, resource items that may need to be singular or plural will have two entries in the resource bundle, one for each case.

In some cases the text information for a particular resource item must include special characters. A line break may be needed for information that is to be shown on multiple lines. A line break is represented as "\n". In general you should not use a line break in a resource item unless the default bundles that are part of the installation also use a line break. While it is acceptable to include multiple, sequential spaces in a resource item, tab characters should not be used. Finally, UNICODE characters outside the normal ASCII range will use a either a four-digit or six-digit hexadecimal code. For example, the letter "ĝ" would be represented in the resource item as "\u011d". For some writing systems such as Japanese, it can be expected that virtually all resource information will consist of UNICODE characters.

Message System Bundles

The textual information used for issuing note, warning, and error messages is organized into three resource bundles. One bundle includes all of the messages that can be generated by the basin model and its components, principally the hydrologic elements. A second bundle holds the messages issued by the meteorologic model. The last bundle contains all of the remaining messages. The default resource bundles are stored in the same directory as the interface resource bundles. They use the same UNICODE specification as discussed earlier.

Each message resource is stored as one line in a resource bundle. The line begins with a unique number to identify the resource item. The number is followed by an equal sign, and then the message text. Following are some sample lines from a resource bundle:

```
40220=No time of concentration set for subbasin \"{0}\".
40221=Invalid time of concentration for subbasin \"{0}\".
40222=No storage coefficient set for subbasin \"{0}\".
40223=Invalid storage coefficient for subbasin \"{0}\".
```

These messages are used when the parameters are checked for the Clark transform method. When the parameter check fails, the program retrieves the correct message and displays it to the user in the message window at the bottom of the main program window.

Messages often include specific information about when something occurred, a value that was out of range, or the name of the component where the error happened. Place holders are used in a message resource item for these items. The program retrieves the message from the resource bundle and then substitutes the correct, specific information into the place holders. In the example lines above, the specific information is the name of the subbasin where the parameter check failed. There may be no place holders, one, or several in a particular message. Each place holder is composed of curly braces and a number indicating the first holder, second, etc.

By convention, named components in a message are enclosed in double quotes. Because a quotation mark is a special character, there must be a back slash before it. Other special characters such as line breaks and UNICODE characters may also be included.

File Conventions and Installation

The program uses a very specific file naming convention for the resource bundles. This convention enables the program to load the correct resource bundles for the locale specified on the computer. The locale is set in the regional or language settings of the

computer, depending on which operating system you use. Resource bundles for alternate languages must begin with the same filename used for the default bundles included in the installation. The alternate bundles must then append to the filename an underscore character and the language code. Finally, a second underscore and the country code are appended after the language code. The following lines show a default resource bundle, along with a bundle for French, and a bundle for French in France:

```
BasinInterfaceResources.properties
BasinInterfaceResources_fr.properties
BasinInterfaceResources_fr_FR.properties
```

Alternate language resource bundles should be placed in the same directory as the default bundles. The program will automatically determine which files to use when it starts based on the locale setting of the computer. If the locale setting changes while the program is running, you will need to close and restart the program to recognize the new locale.

When the program starts, it will automatically detect the locale setting of computer. It will then attempt to load the resource bundle for that specific locale, both the language and country. If the bundle is missing or it is only partially complete, the program will automatically attempt to load the resource bundle for the language that is part of the specified locale. If the program is still not able to find all of the required resource items, it will automatically load the default resource bundles. A partial implementation means that some resource items are translated and placed in appropriate localization resource bundles. A complete implementation means that all resources required by the program can be found in the localized resource bundles.

Setting the locale is specific to the operating system. On computers using Microsoft Windows®, the locale is set in the *Control Panel* under *Regional and Language Options*. An example is shown in Figure 2. The locale is selected from the list that shows language and country combinations.

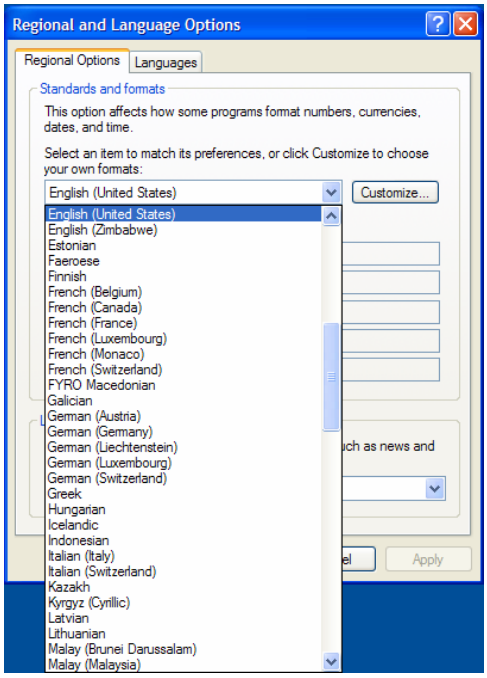

Figure 2. Selecting the locale on a computer using Microsoft Windows®.

Supported Fonts

Most computers include fonts as part of the initial operating system setup. These fonts usually include the basic Latin fonts and any additional fonts necessary for the locale where the setup is performed. For example, a computer setup in the United States is unlikely to have the fonts necessary to display the Thai language. However, a computer setup in Thailand will include the necessary fonts. It is possible to add additional fonts at a later time, either manually or as part of installing a new application.

HEC-HMS does not include any fonts when it is installed. It relies on the operating system to provide the fonts it needs and this will be sufficient if the default resource bundles. Depending on the locale you intend to implement, you may need to make sure that appropriate fonts are present on the computer.

Translation Procedures

The following steps describe the best procedure for implementing the interface in an alternate language.

1. Make a copy of the six resource bundle files. The files are distributed as part of the program installation package. Do not delete or rename the original resource bundles distributed with HEC-HMS; just make the necessary copies. The copies will likely have "read only" file property so you must change them to "read write" in order to continue to the next step.

2. Look up the appropriate code for your locale in Appendix A or Appendix B.

3. Rename the copied files with the correct locale code appended. The code is appended at the end of the filename while the file extension remains unchanged. For example, to produce a resource bundle for German in Austria, the default file:

   ```
   BasinInterfaceResources.properties
   ```

   should be copied and renamed as:

   ```
   BasinInterfaceResources_de_AT.properties
   ```

4. Open the renamed resource bundle in an appropriate file editor program. The program should be able to save the resource file using the UTF-8 or UTF-16 character encoding specification. Your file editor may support many different formats. It is very important to save the file using the UTF-8 or UTF-16 format. HEC-HMS may not work correctly if other file formats are used.

5. Edit each resource item by translating the right side of the equal sign from English to the selected locale language. Do not change the identifier on the left side of the equal sign. Changing the identifier will stop the program from being able to identify resources correctly. For example, the text from the following lines:

   ```
   LAM_Cond_L=Conductivity
   LAM_Cond_TT=Saturated Hydraulic Conductivity
   ```

   would be translated to German as "Leitfähigkeit" and "Gesättigte Hydraulische Leitfähigkeit". The correct text for the resource bundle with UNICODE characters would be:

   ```
   LAM_Cond_L=Leitf\u00E0higkeit
   LAM_Cond_TT=Ges\u00E0ttigte Hydraulische Leitf\u00E0higkeit
   ```

6. Delete any lines from the copied resource bundles that you do not translate. This will allow the program to automatically load those resources from the default bundles. All information loaded from the default bundles is displayed in English regardless of the locale setting on the computer.

7. Place the translated resource bundles in the installation directory along with the default bundles.

8. Start the program to see the implementation of your work.

Program Releases and Versioning

Each version of HEC-HMS is installed separately on a computer. This means that each version is completely self-contained and does not share resources with other versions which may also be installed. Translated resource bundles must also be installed separately for each version of the program.

The default resource bundles may change for each new program release even though we make every effort to keep the bundles as stable as possible. We try not to change identifiers used to find text in the resource bundles. For example, if a menu command uses the identifier `ED_All_L` to retrieve the text resource for the "Select All" command on the "Edit" menu, we try to avoid changing the identifier to something else in a future release version. However, identifier changes are sometimes necessary to improve organization of the resource bundles. We also try to avoid changing the text resource that is connected to a particular identifier. For example, we would try to avoid changing the displayed name of the command from "Select All" to something else such as "Select All Elements". However, sometimes we must change text resources to improve usability.

New program releases will generally require a new set of translated resource bundles. There may be changes to identifiers or text resources. Because of new features in the program, new resources will be added to the bundles. File comparison programs (professional or open source) are available that can be used to show differences between a default resource bundle for two different program releases. The comparison tool will show resource identifiers that are only in the resource bundle from the older program release. The resources can be safely removed from the bundle you prepare for the new release. The tool will also show resource identifiers that are only in the bundle for the new release. These resources are new and must be added to your translated bundle. The tool can also show any identifiers that have changed spelling or any text resources that have changed. Any resources that have not changed can be used in the new resource bundle exactly as you prepared them for the old bundle.

After you have identified changes in the default resource bundles from one program release to the next release, you can use the following steps to update your translated bundles.

1. Start with the translated bundles you prepared for the old program release. Make a copy for use with the new release.

2. Edit the new copy to update any resource identifiers that have changed. Update the translated text for any identifiers where we changed the default English text.

3. Add identifiers and translated text for any new resources.

4. Finally you may distribute the update resource bundles for the new HEC-HMS release by placing them in the appropriate installation directory.

Appendix A – Fully Supported Locales

Language	Country	Locale ID
Arabic	Saudi Arabia	ar_SA
Chinese (Simplified)	China	zh_CN
Chinese (Traditional)	Taiwan	zh_TW
Dutch	Netherlands	nl_NL
English	Australia	en_AU
English	Canada	en_CA
English	United Kingdom	en_GB
English	United States	en_US
French	Canada	fr_CA
French	France	fr_FR
German	Germany	de_DE
Hebrew	Israel	iw_IL
Hindi	India	hi_IN
Italian	Italy	it_IT
Japanese	Japan	ja_JP
Korean	South Korea	ko_KR
Portuguese	Brazil	pt_BR
Spanish	Spain	es_ES
Swedish	Sweden	sv_SE
Thai (Western digits)	Thailand	th_TH
Thai (Thai digits)	Thailand	th_TH_TH

Appendix B – Provided But Untested Locales

Language	Country	Locale ID
Albanian	Albania	sq_AL
Arabic	Algeria	ar_DZ
Arabic	Bahrain	ar_BH
Arabic	Egypt	ar_EG
Arabic	Iraq	ar_IQ
Arabic	Jordan	ar_JO
Arabic	Kuwait	ar_KW
Arabic	Lebanon	ar_LB
Arabic	Libya	ar_LY
Arabic	Morocco	ar_MA
Arabic	Oman	ar_OM
Arabic	Qatar	ar_QA
Arabic	Sudan	ar_SD
Arabic	Syria	ar_SY
Arabic	Tunisia	ar_TN
Arabic	UAE	ar_AE
Arabic	Yemen	ar_YE
Belorussian	Belorussia	be_BY
Bulgarian	Bulgaria	bg_BG
Catalan	Spain	ca_ES
Chinese	Hong Kong	zh_HK
Croatian	Croatia	hr_HR
Czech	Czech Republic	cs_CZ
Danish	Denmark	da_DK
Dutch	Belgium	nl_BE
English	India	en_IN
English	Ireland	en_IE
English	New Zealand	en_NZ
English	South Africa	en_ZA
Estonian	Estonia	et_EE
Finnish	Finland	fi_FI
French	Belgium	fr_BE
French	Luxembourg	fr_LU

Provided But Untested Locales – Continued

Language	Country	Locale ID
French	Switzerland	fr_CH
German	Austria	de_AT
German	Luxembourg	de_LU
German	Switzerland	de_CH
Greek	Greece	el_GR
Hungarian	Hungary	hu_HU
Icelandic	Iceland	is_IS
Italian	Switzerland	it_CH
Latvian	Latvia	lv_LV
Lithuanian	Lithuania	lt_LT
Macedonian	Macedonia	mk_MK
Norwegian (Bokmål)	Norway	no_NO
Norwegian (Nynorsk)	Norway	no_NO_NY
Polish	Poland	pl_PL
Portuguese	Portugal	pt_PT
Romanian	Romania	ro_RO
Russian	Russia	ru_RU
Slovak	Slovakia	sk_SK
Slovenian	Slovenia	sl_SI
Spanish	Argentina	es_AR
Spanish	Bolivia	es_BO
Spanish	Chile	es_CL
Spanish	Colombia	es_CO
Spanish	Costa Rica	es_CR
Spanish	Dominican Republic	es_DO
Spanish	Ecuador	es_EC
Spanish	El Salvador	es_SV
Spanish	Guatemala	es_GT
Spanish	Honduras	es_HN
Spanish	Mexico	es_MX
Spanish	Nicaragua	es_NI
Spanish	Panama	es_PA

Provided But Untested Locales – Continued

Language	Country	Locale ID
Spanish	Paraguay	es_PY
Spanish	Peru	es_PE
Spanish	Puerto Rico	es_PR
Spanish	Uruguay	es_UY
Spanish	Venezuela	es_VE
Turkish	Turkey	tr_TR
Ukrainian	Ukraine	uk_UA
Vietnamese	Vietnam	vi_VN