Parlez-Vous...

By Bill Bergmann

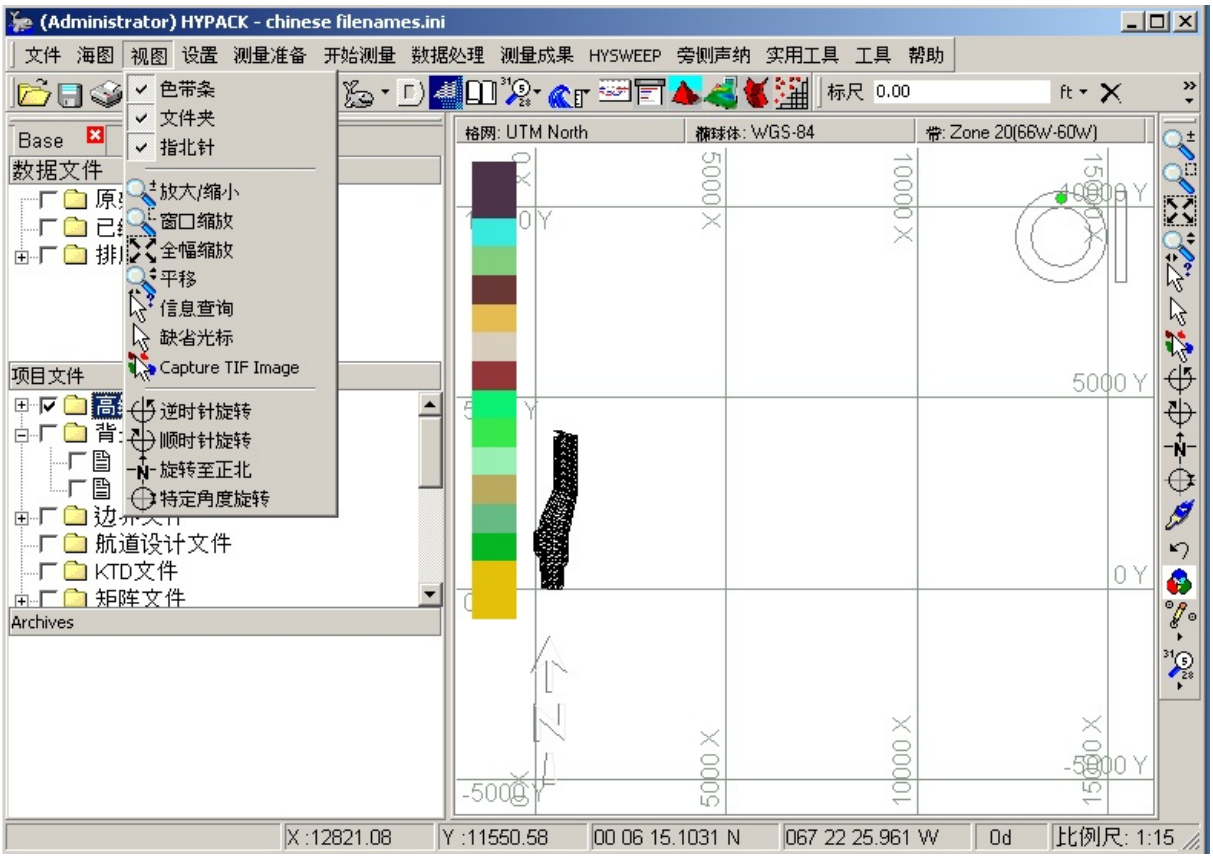
In an ongoing effort to better support our international customers, HYPACK, Inc. is in the process of upgrading its software to support Unicode. This is a standard, developed and maintained by the The Unicode® Consortium, which supports, in theory, every known language and writing system in use today as well as many dead languages and a plethora of obscure symbols. It still has space for future additions, as it can store over a million entries of which, as of a recent post I saw, only some 94,000 entries are currently used. This is quite a change from the 255 symbols available in the ANSI character set most of us in the Western world were brought up on.

The majority of computer software has been developed based on an 8-bit character size, thus giving the 255 entries noted above. In order to support different languages, those same bits could be interpreted in differing ways. That is to say, the data was encoded for a specific language, like English or Cyrillic. The encoding thus controlled the meaning of the entry at certain positions in the set. For example the entry at position 65 means the letter 'A' in one system or perhaps an umlaut in another. The Code Page identifier is the key which told software how the characters in the set should be interpreted. Having a suitable font available, which contains either a 'picture' of the symbol or drawing commands to form the symbol, completes the visual presentation for the user.

Sound like a good system? Well, it was anything but. Code pages were popping up all over the place, IBM making this one, Microsoft another, someone's Uncle yet another and so on. Without consistency and an authoritative source to rely on, it was nearly impossible for software to support multiple languages, let alone transfer information between systems. I am sure we have all run across 'garbage' on the screen at least once when trying to view some text. The cause is more likely than not the software interpreting the data with the wrong code page or missing a font to support the symbols the encoder intended.

HYPACK® 2011 is hopefully the last one to rely on code pages! The shell for HYPACK® 2012 will display a user's native language selection without depending on a version of the operating system with special support installed for it.

FIGURE 1. *HYPACK® Chinese Language Interface*



Switching to the Unicode model has solved another problem which was not addressable before. It is perfectly natural to give meaningful names to your data files, and if you are, for example, native Chinese speaker, naming them with Chinese characters would be a reasonable thing to do. Given that today's operating systems have inherently supported Unicode for many years, the capability of doing just such a thing is common place. Unfortunately, getting a file name encoded in Unicode into a program that doesn't have Unicode support is basically impossible. For instance, the File Open dialog will allow one to select a Unicode file, but when the software asks for the name, it receives a series of question marks. It was just such a bug report from one of our customers which finally got us moving in this direction.

Now one might ask, "What happens to my data files, like the project file, log files, etc. when you store Unicode text inside of them? Will they be incompatible with programs that are still Unicode illiterate?" The answer is good news all around. The Unicode standard prescribes 3 encoding methods which can all perfectly describe any entry in Unicode. These are based on the byte sizes 8, 16 and 32. To interact with the Microsoft O/S, data is stored internally in UTF-16. UTF stands for Universal Translation Format. UTF-16 requires (nearly all the time) exactly twice as much space as the old 8 bit code page model. UTF-8, as the name implies, is based on 1 byte or 8 bits per character. It is has, by design, the useful property of being identical to, in the first 255 positions, the 8 bit Ansi format! This is great news. By encoding into UTF-8 when writing files, older software will not break. It may not be able to use all information in the file, but it won't break. A boon for backward compatibility. If you work in

English only, you will probably never even notice that Unicode is working inside the program behind the scenes.

There are, of course, many gory details I have left out. If you are very curious, an internet search can quickly bring up some documents to put even the worst insomniac to sleep... However, if you are a typical user, you just expect the software to work, and I for one don't blame you. That's my viewpoint also.