

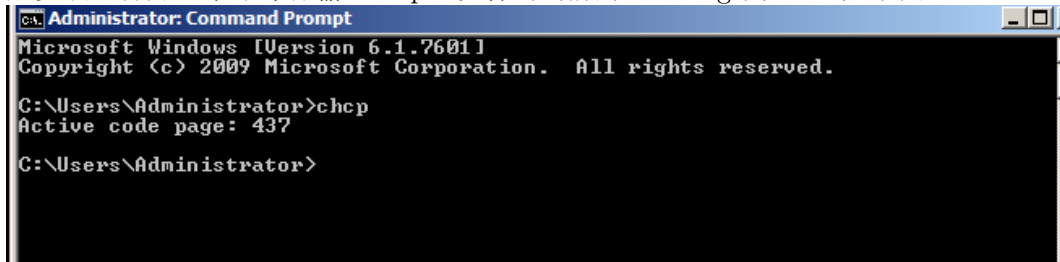
Ovation的中文编码分析

ZW

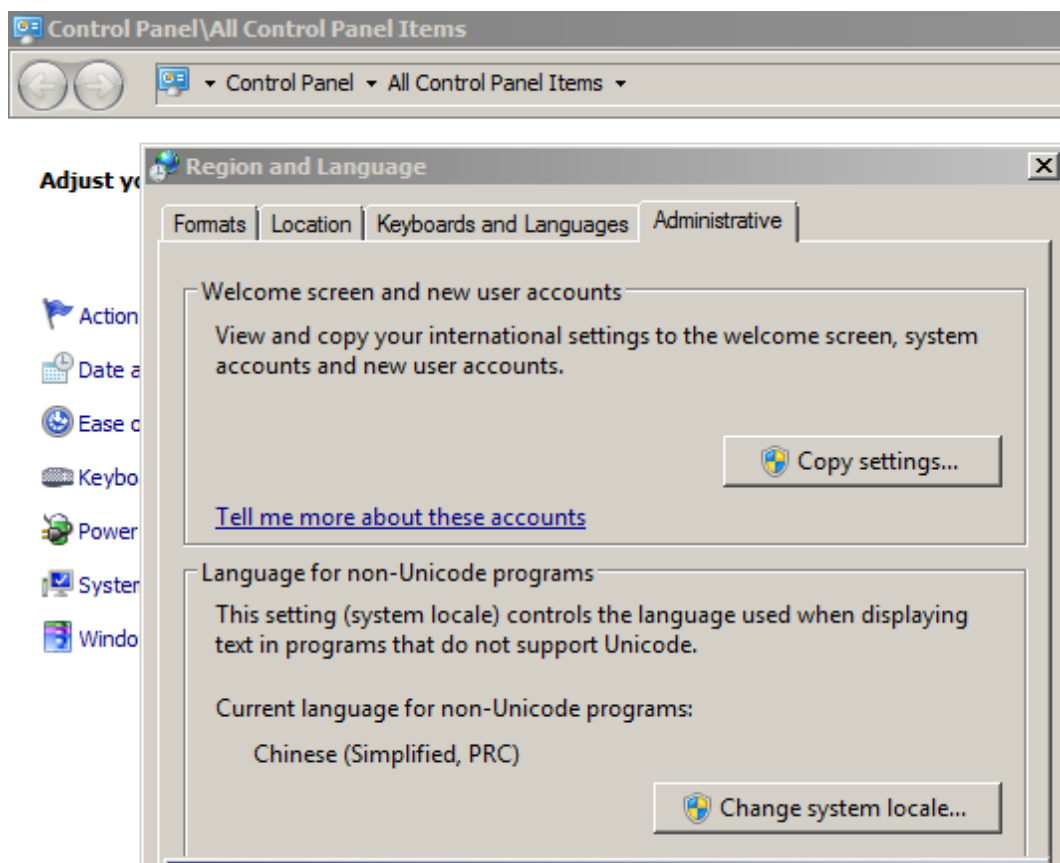
2022年3月 v1.0

1 中文编码概述

在Windows中字符以字节方式存储，最基本的ASCII编码使用一个字节存储一个字符，ASCII字符从0x00-0x7F，实际上只用了7位，占用一个8位的字节。Ovation使用Oracle数据库存储数据，Oracle字符集为AMERICAN_AMERICA.WE8MSWIN1252，字符存储在VARCHAR2类型中。Ovation中的中文显示就依赖于CodePage的设置，如果安装英文版本的Windows，刚完成安装，打开一个命令行输入chcp可以看到当前的CodePage为437，如下图：

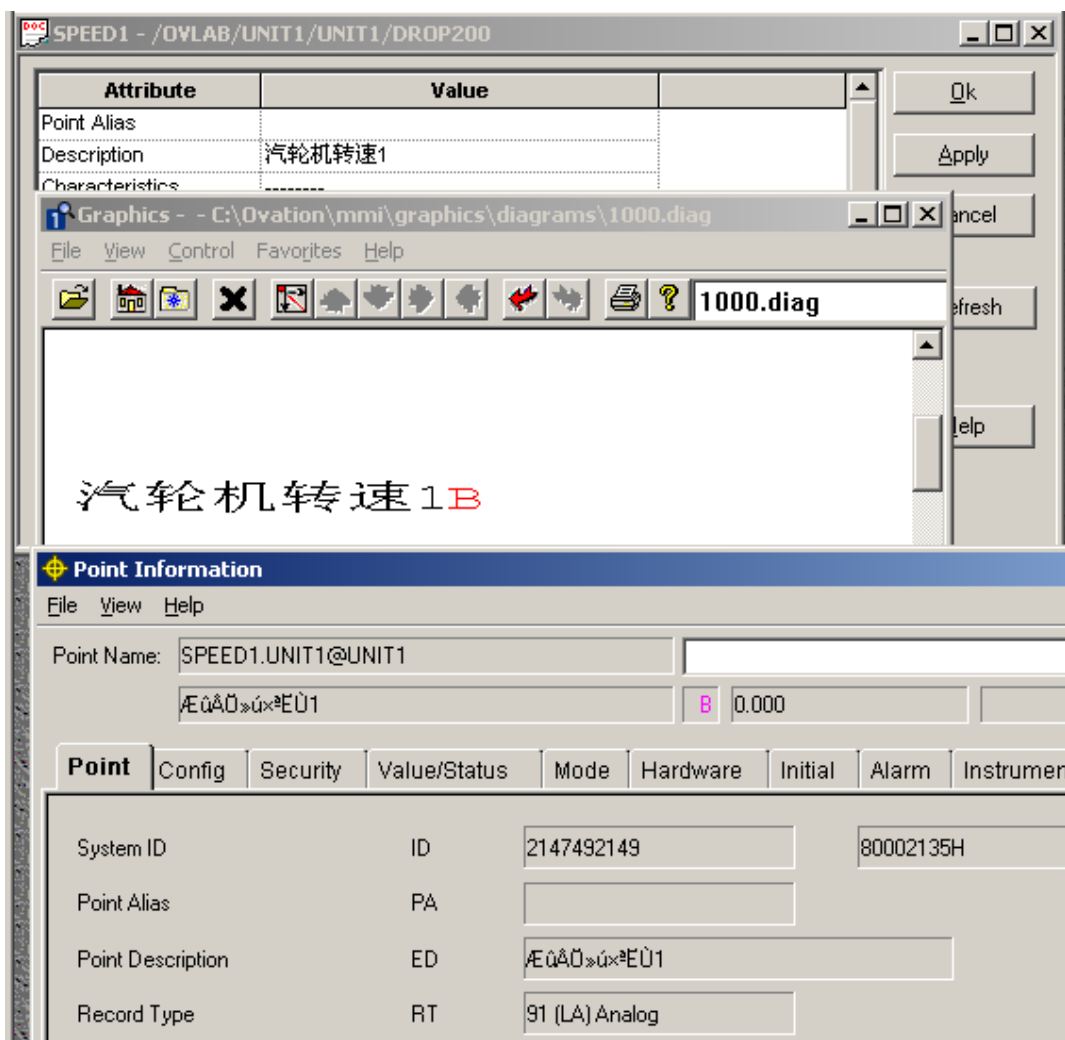
A screenshot of a Windows Command Prompt window titled "Administrator: Command Prompt". The window shows the following text: "Microsoft Windows [Version 6.1.7601] Copyright (c) 2009 Microsoft Corporation. All rights reserved. C:\Users\Administrator>chcp Active code page: 437 C:\Users\Administrator>". The text is displayed in a black monospace font on a white background.

如果要修改系统的CodePage，只需要在Control Panel的Region and Language中修改设置就可以，如下图那样修改为Chinese(Simplified,PRC)，修改为简体中文并重启电脑后，CodePage就变为936：

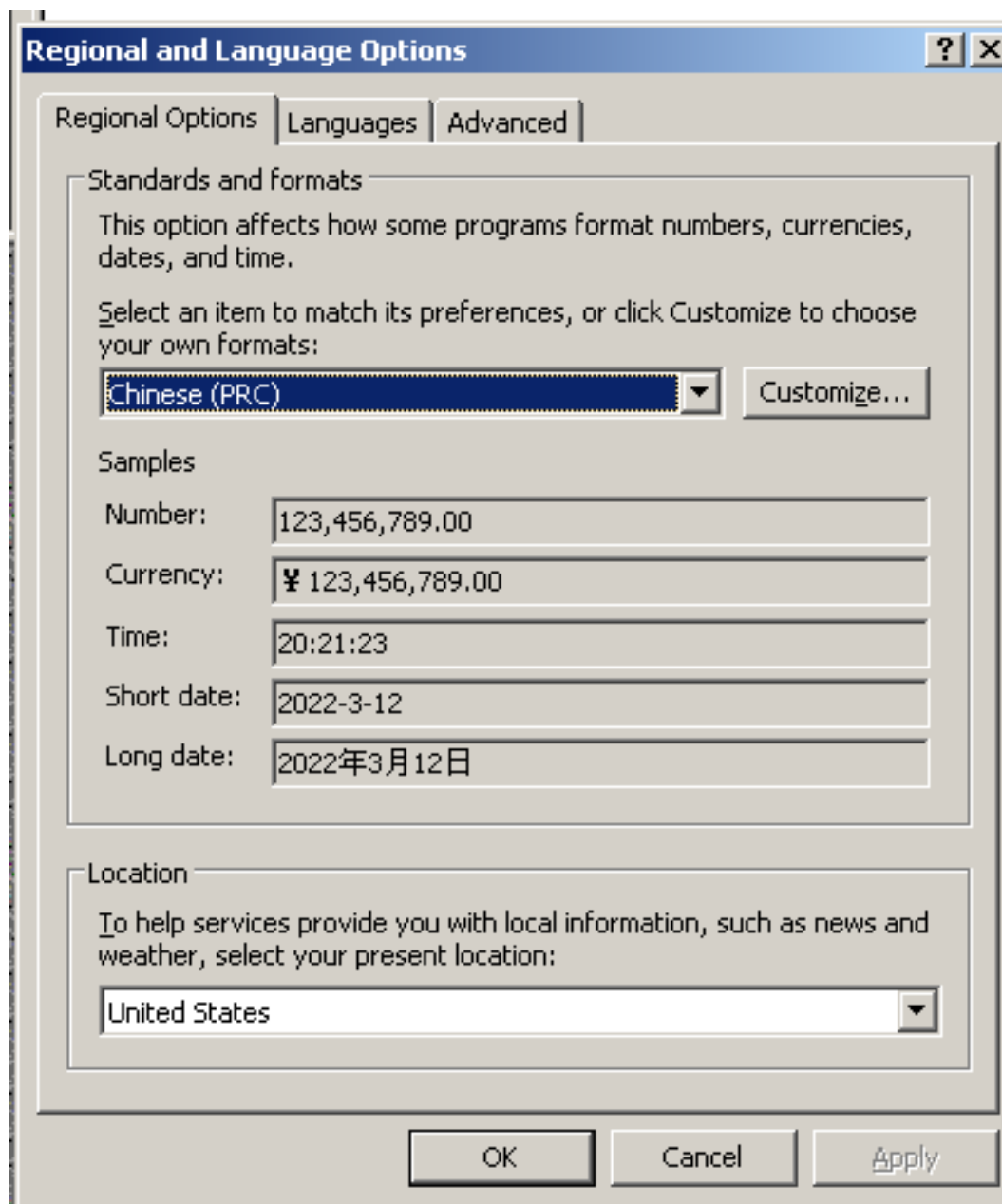


这个时候我们就可以在Ovation Studio中输入中文了，在Ovation的Oracle数据库中的中文存储按照GB2312/GBK/GB18030编码存储。

先来举个例子吧，在Studio中建立一个SPEED1的LA点，描述为“汽轮机转速1”。在Ovation 3.2中，完成这个点的Load后，在Graphics和Point Information中查看这个点，如图：



我们看到Ovation Studio和Graphics都能准确显示中文，而Point Information显示乱码。这个时候我们就要再到Control Panel->Regional and Language Option的第一个下拉菜单改为Chinese(PRC)。



然后再把Point Information关闭后，重新打开就会显示正常的中文了。接下来我们把分数据库用OvPtExport导出后，看看中文是如何存储的。使用vim打开分数据库的备份文件，转换成16进制显示，然后看SPEED1这个点的Description，如下图：

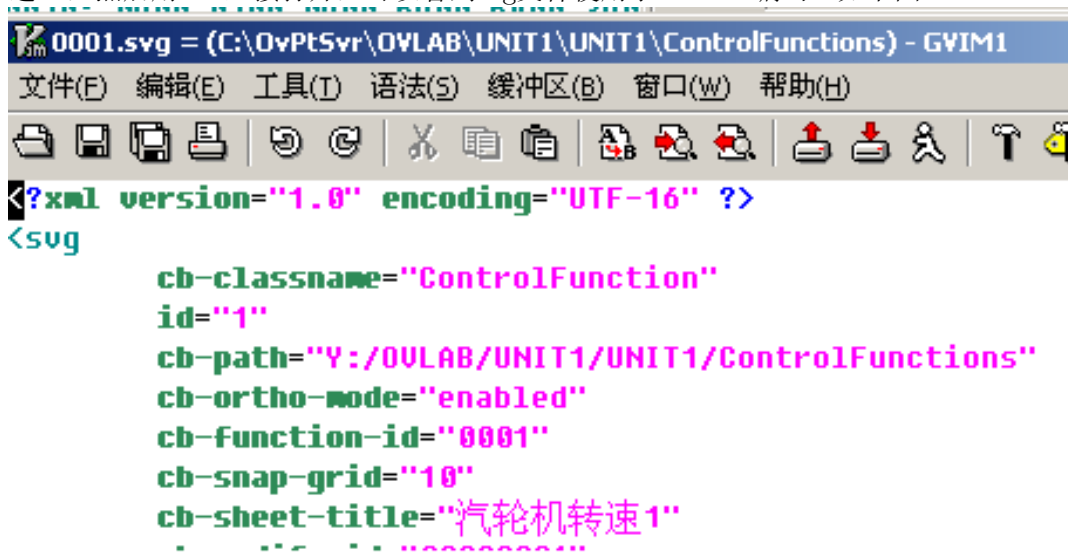
```

30021c70: 5045 3d22 416e 616c 6f67 506f 696e 7422 PE="AnalogPoint"
30021c80: 204e 414d 453d 2253 5045 4544 3122 0d0a NAME="SPEED1"..
30021c90: 2020 2020 205b 4445 5343 5249 5054 494f [DESCRIPTIO
30021ca0: 4e3d 22c6 fbc2 d6bb fad7 aacb d931 220d N=".....1".
30021cb0: 0a20 2020 2020 4252 4f41 4443 4153 545f . BROADCAST_
30021cc0: 4652 4551 5545 4e43 593d 2253 220d 0a20 FREQUENCY="S"..
30021cd0: 2020 2020 4f50 505f 5241 5445 3d22 5322 OPP_RATE="S"
30021ce0: 0d0a 2020 2020 2043 4841 5241 4354 4552 .. CHARACTER

```

可以看到中文字符“汽轮机转速1”的实际的存储编码是“C6 FB C2 D6 BB FA D7 AA CB D9 31”，其中最后的16进制0x31就是ASCII字符“1”，而其他10个字节正好两个字节表示一个中文。这里简单介绍一下GB2312编码，GB2312编码收录6763个汉字，用两个字节表示一个汉字。GBK为汉字内码扩展规范，收录汉字（包括部首和构件）21003个，图形符号883个。GB 18030，全称为信息技术中文编码字符集，共收录汉字70244个，对GB2312-1980完全向后兼容，与GBK基本向后兼容。“汽”这个汉字在GB2312中的区位码是(38,91)，即38区91位，存储时以EUC-CN方式存储，高低位都要加上0xA0(160)，这样实际的存储就是0xC6,0xFB了。GBK和GB18030中这个汉字也是同样的存储。那么我们刚才看到的乱码是怎么来的呢？实际上是系统将两个字节的汉字码解释为单个字节的ASCII码造成的，由于这里的码都超过了0x7F，实际的字符是扩展ASCII码中的，这样这10个字节按照扩展ASCII表示就成了这样的乱码：ÆûÂÖ»ú×ªËÛ。

接下来我们再来看看Control Builder的情况，我们新建一个逻辑图，取名为“汽轮机转速1”，然后用vim直接打开，可以看到svg文件使用了UTF-16编码：如下图：



```

0001.svg = (C:\OvPtSvr\OVLAB\UNIT1\UNIT1\ControlFunctions) - GVIM1
文件(E) 编辑(E) 工具(T) 语法(S) 缓冲区(B) 窗口(W) 帮助(H)
<?xml version="1.0" encoding="UTF-16" ?>
<svg
    cb-classname="ControlFunction"
    id="1"
    cb-path="Y:/OVLAB/UNIT1/UNIT1/ControlFunctions"
    cb-ortho-mode="enabled"
    cb-function-id="0001"
    cb-snap-grid="10"
    cb-sheet-title="汽轮机转速1"

```

接着用vim转换为16进制显示，如下图：

```

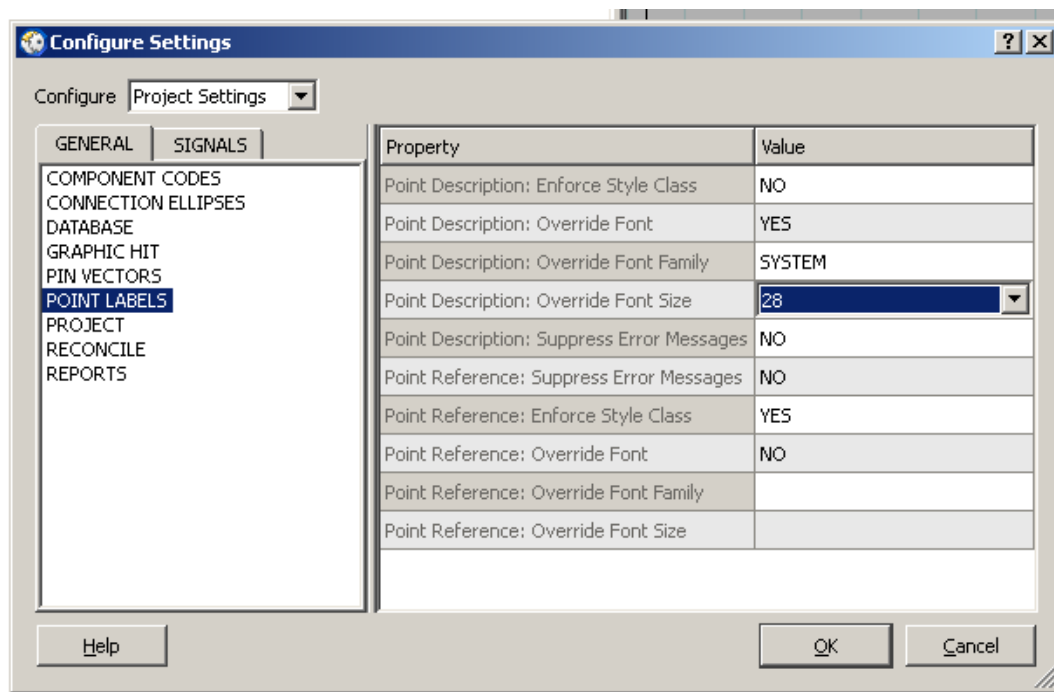
00000190: 6700 7200 6900 6400 3d00 2200 3100 3000 g.r.i.d.=."1.0.
000001a0: 2200 0d00 0a00 0900 6300 6200 2d00 7300 ".....c.b.-.s.
000001b0: 6800 6500 6500 7400 2d00 7400 6900 7400 h.e.e.t.-.t.i.t.
000001c0: 6c00 6500 3d00 2200 7d6c 6e8f 3a67 6c8f l.e.=.">
000001d0: 1f90 3100 2200 0d00 0a00 0900 6300 6200 ..1.".....c.b.
000001e0: 2d00 6d00 6f00 6400 6900 6600 7900 2d00 -.m.o.d.i.f.y.-.
000001f0: 6900 6400 3d00 2200 3000 3000 3000 3000 i.d.=."0.0.0.0.
00000200: 3000 3000 3000 3100 2200 0d00 0a00 0900 0.0.0.1.".....
00000210: 6800 6500 6900 6700 6800 7400 3d00 2200 h.e.i.g.h.t.=."

```

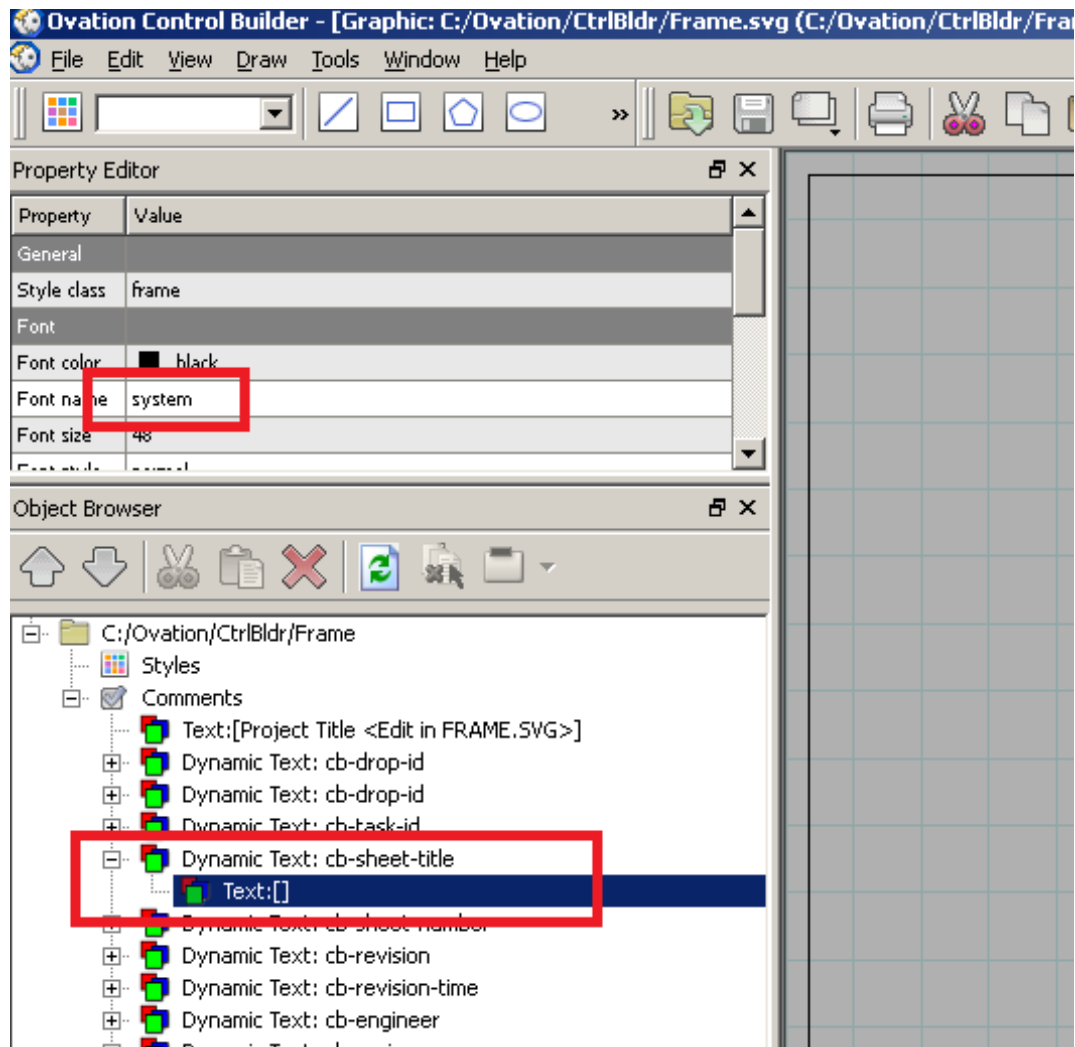
我们可以看到在UTF-16模式下，这几个中文字符以及字符1都是每个字符两个字节表示，但是他们的编码又都和GB编码不同，“汽”字在UTF-16编码为0x7D 0x6C。如果我们这时候将系统切换回英文，会发现中文又出现了乱码（在Control Builder和Ovation Studio中的显示一样），而且这些中文是先被从UTF-16转换为GB编码，再以扩展ASCII表示，这样就变成了和之前在Point Information看到的一样的乱码。不过如果是SVG图纸中的普通Text，只要系统安装了字库就会正确显示。这是因为点的描述每次打开Control Builder时如下图：



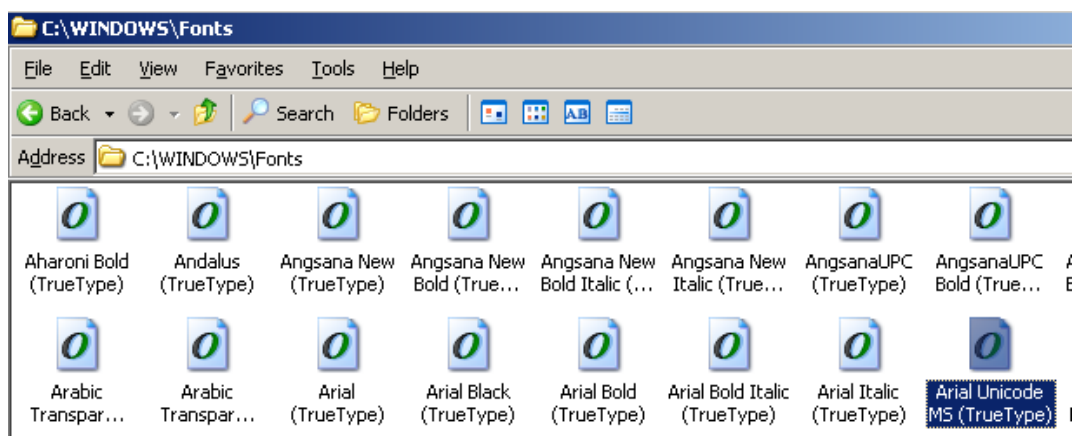
对于Control Builder的中文显示除了正常的系统语言设置，还需要在Control Builder的菜单的Editor->Configuration->GENERAL->POINT LABELS做相应设置，如下图：



另外图纸的右下角的标题需要正确显示中文的话，需要打开C:\Ovation\CtrlBldr\Frame.svg修改cb-sheet-title的字体设置，从默认的Arial修改为System字体，如下图：



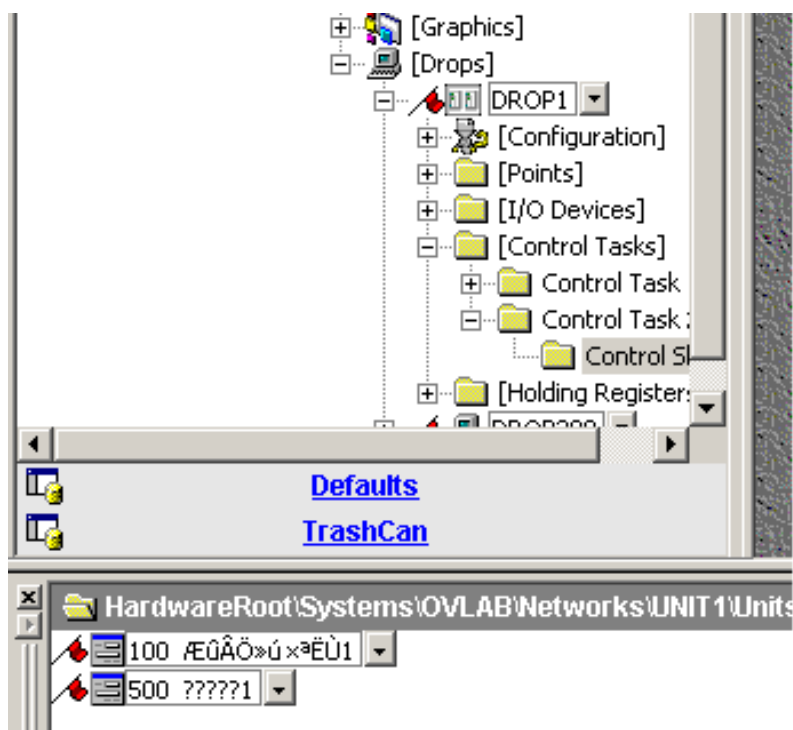
如果我们还需要把图纸Publish输出为PDF文档，需要在Windows的字体中安装ArialUnicode字体，这是因为Publish时的文本使用了ArialUnicode字体，如下图：



2 备份导入

当我们用备份恢复系统的时候，语言环境的设置有没有影响呢？

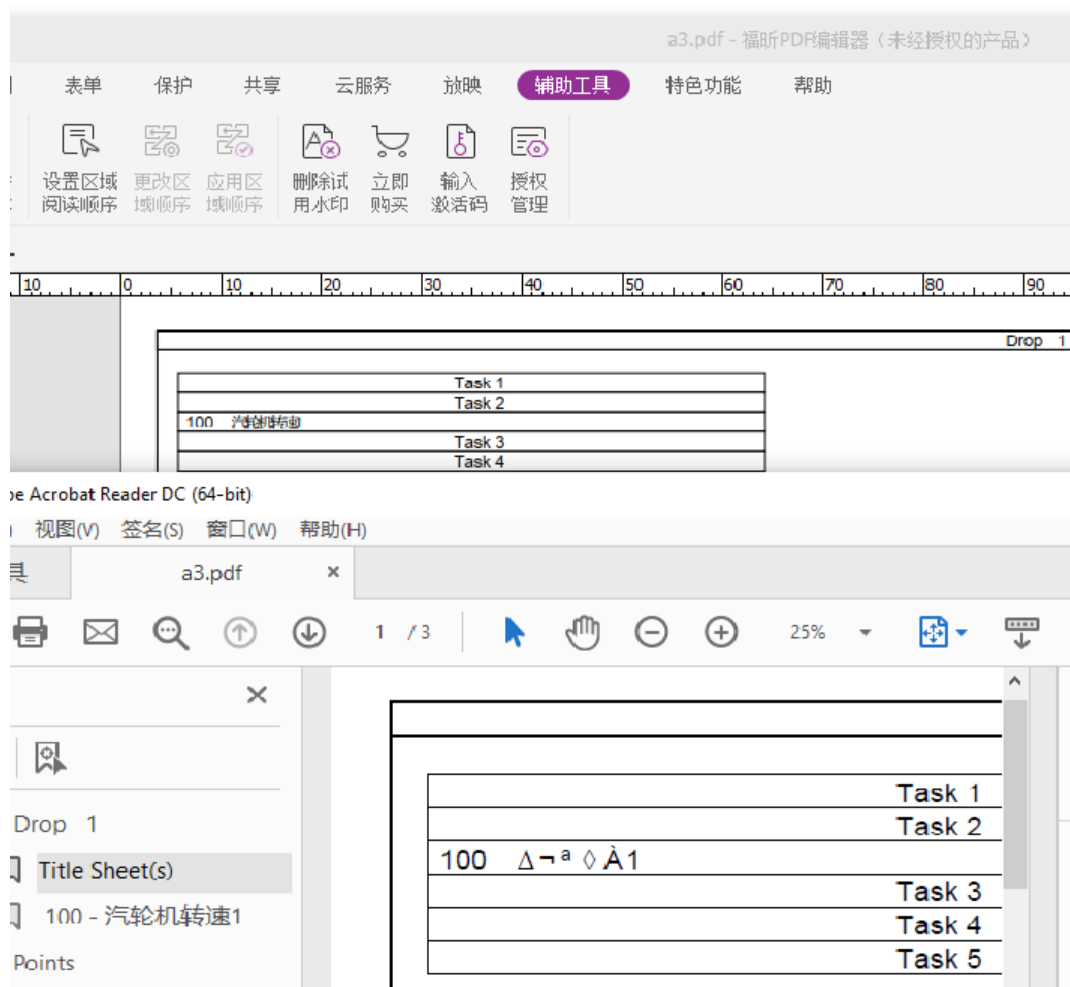
(1)全数据库恢复，使用do_ptadmin_import.bat脚本恢复全数据库备份，脚本使用Oracle的imp命令导入备份，不受语言环境设置的影响。完成数据库的导入后，直接覆盖整个OvptSvr目录就可以恢复系统。不需要编译逻辑图和画面，当然编译逻辑图和画面也无关语言环境设置，但是如果要导入中文Title的逻辑图，必须要切换语言环境，否则在Studio中就会出现乱码Title，而这种带有问号的Title是被破坏的中文，无法恢复。如下图：



(2)分数库恢复，使用ptdbimport导入分数数据库备份，实际上也不受语言环境设置的影响。也就是说用ptdbimport单独导入测点，也是不受系统语言环境的影响。需要注意的是，导入逻辑图时如果Title有中文就必须切换切换到中文语言环境。画面的导入则不受语言环境设置的影响。

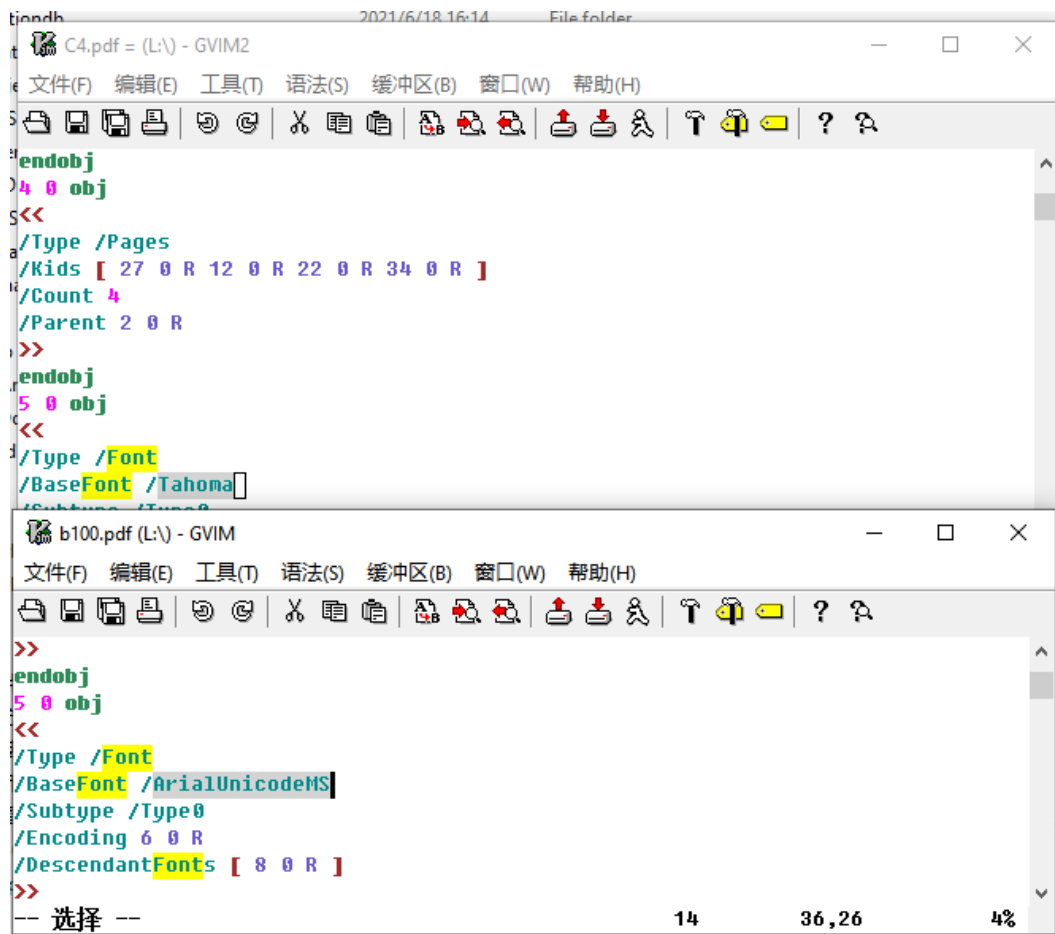
3 CB的Publish的相关问题

前面我们说了Control Builder的Publish需要的设置，字体等问题。这里还要说一下PDF阅读器的问题。通常推荐使用的PDF阅读器有Acrobat Reader，Foxit Reader或者Microsoft Edge浏览器。对于Ovation 3.2版本，会发现每个控制器的目录页的中文，使用Acrobat/Edge是无法正确浏览的，用Foxit则没问题，不过总的来说目录页也并不重要，因为侧边的目录的中文都是没问题的，如下图：



虽然Foxit Reader可以正确显示目录页的中文，但是如果在逻辑图中自行添加说明文本时，没有将字体改为Arial Unicode MS，则Foxit Reader就无法看到这些文本，而Acrobat/Edge则不受字体定义的影响。

Ovation3.8开始不知道为什么把Publish的字体由ArialUnicodeMS改为了Tahoma，造成Acrobat/Foxit都没法看导出pdf中的中文了，不过Edge倒是可以。还有一种办法就是用vim直接修改导出pdf中的BaseFont，将Tahoma改为ArialUnicodeMS，有两处需要修改，搜索一下就可，改完后Acrobat和Foxit也可以正确显示中文。如下图：



```
endobj
4 0 obj
<<
  /Type /Pages
  /Kids [ 27 0 R 12 0 R 22 0 R 34 0 R ]
  /Count 4
  /Parent 2 0 R
>>
endobj
5 0 obj
<<
  /Type /Font
  /BaseFont /Tahoma
  /Subtype /Type0
>>
endobj
5 0 obj
<<
  /Type /Font
  /BaseFont /ArialUnicodeMS
  /Subtype /Type0
  /Encoding 6 0 R
  /DescendantFonts [ 8 0 R ]
>>
-- 选择 --
```

14 36,26 4%

... Ending。